Servicios en red Documentation

Versión 2017.0

José Domingo Muñoz

19 de septiembre de 2018

Índice general

El módulo profesional de **Servicios de Red e Internet** se imparte durante el segundo curso del Ciclo Formativo de Grado Superior de Administración de Sistemas Informáticos en Red (ASIR).

De acuerdo a la normativa reguladora del ciclo formativo, el módulo profesional de Servicios en red se imparte durante el segundo curso y tiene asignadas un total de 126 horas, a razón de 6 horas semanales durante 21 semanas.

El índice de contenidos que vamos a estudiar será:

CAPÍTULO 1

Introducción a los servicios en red

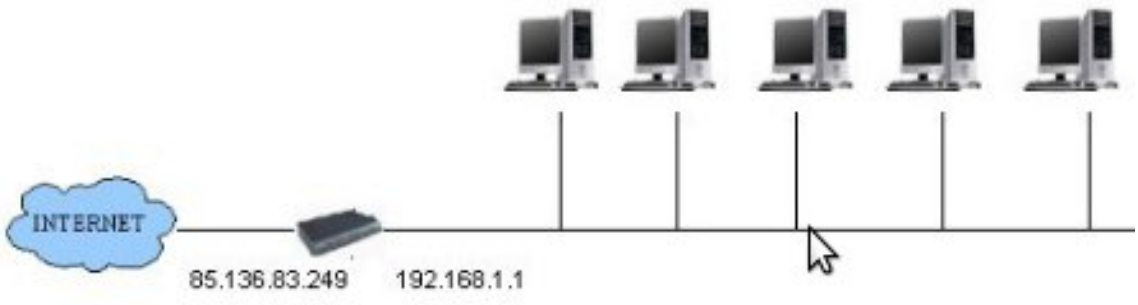
En este primer apartado vamos a estudiar los siguientes apartados:

- ¿Qué son lo servicios de red?
- Vamos a recordar conceptos sobre redes, TCP/IP, DNS,...
- Python como lenguaje para el administrador de SI
- Herramienta (vagrant) que nos va a permitir trabajar con nuestra infraestructura.

1.1 Cuestionario de repaso

1.1.1 Repaso TCP/IP

1. Teniendo en cuenta el siguiente esquema de red:



   - Configura la interfaz de red de un cliente para tener acceso a internet, utiliza direccionamiento estático. ¿Qué diferencia hay entre dirección estática y dinámica? ¿Qué dirección del router es la pública? ¿Cuál es la privada? Define cada uno de los parámetros que has configurado: puerta de enlace, mascara de red,...

2. Cambia el direccionamiento de red de nuestra internet con la red 172.22.0.0/16 ¿Cuantos equipos podemos tener en esta red?
3. Nuestro clientes pueden acceder a internet porque el router hace Source NAT ¿Explica en que consiste? Si tenemos un servidor linux haciendo de router, ¿cómo se configura para hacer SNAT?. Del mismo modo si tenemos un servidor Windows, ¿cómo se se configura?
4. ¿Qué puerto se utiliza por defecto para conectarse a un servidor web?¿Y para el servidor DNS? ¿Para qué se usa el puerto 22?
5. Imagina que en nuestra intranet instalamos un servidor web. ¿Qué configuración hay que hacer en el router para poder acceder desde internet al servidor? ¿Cómo se llama esta técnica?
6. Si nombramos las máquinas de nuestra intranet, y tenemos un sistema linux, ¿en qué fichero se configura el nombre?
7. ¿Qué es la resolución estática de nombres? ¿En qué fichero se configura en Windows?¿Y en linux?
8. Si nuestro cliente tiene un sistema linux, ¿en qué fichero hemos configurado la red?¿y los servidores DNS?
9. Muestra la configuración de red de tu ordenador de clase. ¿Qué servidor DNS se está utilizando.

1.1.2 Repaso DNS

1. ¿Qué herramientas se usa en linux para realizar una consulta DNS?¿Y en Windows? Pregunta en varios sistemas cuál es la dirección IP de www.marca.com. Realiza una consulta para saber los servidores DNS que conocen el dominio gonzalonazareno.org.
2. ¿Qué ocurre si hacemos una consulta para averiguar la ip de dit.gonzalonazareno.org desde el ordenador del aula y desde el ordenador de tu casa? Razona la respuesta.
3. A qué servidor DNS le estás consultando desde clase. Realiza una consulta a www.google.es consultando a nuestro DNS. Vuelve a hacer la consulta usando el servidor público que ofrece google.
4. ¿Qué información puedo guardar en una zona DNS? ¿Qué registros puedo guardar? ¿Cuantos tipos de zonas existen?
5. Si desde clase, consulto la dirección IP de www.josedomingo.org? ¿Cuál es el proceso de consultas que se realiza? ¿A que servidores se va preguntando?
6. ¿Qué son los root server? ¿Cuántos hay?
7. Haz una consulta a www.nyu.edu ¿Cuánto ha tardado la consulta?. Vuelve a hacerla, ¿Ha durado menos? ¿por qué?
8. ¿Por qué desde clase la consulta a la dirección IP dit.gonzalonazareno.org es distinta que si la hacemos desde casa?
9. ¿Qué dirección IP tiene babuino.gonzalonazareno.org y dit.gonzalonazareno.org desde clase?¿Qué relación existe ambos nombres?
10. ¿A que servidor mandamos el correo cuya dirección de destino es correo@josedomingo.org? ¿Y si lo mandamos a usuario@amazon.es?
11. Desde clase, ¿cómo se llama el ordenador que tiene la dirección IP 192.168.103.2?
12. ¿Por qué hay varios servidores con autoridad sobre la zona josedomingo.org?
13. Una vez que sepas la dirección del servidor con autoridad sobre la zona josedomingo.org, realiza una consulta a ese servidor preguntando por www.josedomingo.org. ¿qué proceso de consultas se sigue?

1.2 Python para sysadmins

Para automatizar muchas de las tareas que realizan los administrador de sistemas es necesario crear scripts. Éstos se pueden crear en distintos lenguajes de programación, por ejemplo bash. En este apartado vamos a introducir las posibilidades que nos ofrece python para crear script de administración.

Nota: Si quieres más información puedes consultar la página Python for system administrators.

1.2.1 Resumen de instrucciones

Veamos algunas instrucciones que nos pueden ayudar en nuestrs scripts de administración:

- **Trabajar con argumentos en la línea de comandos**: La mayoría de los script que realicemos recibiran la entrada por argumentos de la línea de comandos:

```
import sys
len (sys.argv) #Número de argumentos
sys.argv[1] #Acceso al segundo argumento
```

- **Salir del programa**: Nos puede interesar que el programa termine baja alguna circunstancia:

```
import sys
sys.exit(0)
```

- **La libreía os te permite acceder del sistema operativo**: Esta librería es muy importante para realizar scripts de administración, veamos algunos ejemplos:

```
import os
print os.getcwd() #Devuelve el directorio donde estás trabajando
os.chdir("Descargas") #Cambia de directorio
os.system("clear") #Ejecuta una instrucción pero no podemos obtener el resultado
```

- **Ejecutar una instrucción y obtener el resultado**: Tenemos tres posibilidades:

```
#Utilizando la librería commands
import commands
data = commands.getoutput("ls -l")
type(data)
lineas=data.split("\n")

#Otra manera, pero poco "elegante"
os.system("ls -l>tmp.txt")
f=open("tmp.txt","r")

#Ejecutar la instrucción como si fuera un fichero
f=popen("ls -l","r")
```

1.2.2 Ejercicios

Realiza un script en python que realice la siguiente función:

1. Muestra el directorio de trabajo.

2. Muestra cuantos usuarios hay definido en el sistema
3. Muestra los usuarios conectados al equipo.
4. Script que lea el nombre de un usuario, si existe dice si es administrador o no, si no existe da un mensaje de error. Realiza el script leyendo el usuario por teclado, y realiza otra versión indicándolo como parámetro.
5. Pasa por parámetros una dirección ip y un nombre de máquina e inserta en `/etc/hosts` (en la tercera línea) la resolución estática. Si no se introducen dos parámetros se da un error.
6. Para crear un usuario «a mano»:
   - Editar `/etc/passwd` y agregar una nueva linea por cada nueva cuenta. Teniendo cuidado con la sintaxis. Debería hacer que el campo de la contraseña sea "*", de esta forma es imposible ingresar al sistema.
   - De forma similar, edite `/etc/group` para crear también un grupo.
   - Crea el directorio Inicio del usuario con el comando *mkdir*.
   - Copia los archivos de `/etc/skel` al nuevo directorio creado
   - Corrige la pertenencia del dueño y permisos con los comandos *chown* y *chmod* (Ver paginas de manual de los respectivos comandos). La opción `-R` es muy útil. Los permisos correctos varían un poco de un sitio a otro, pero generalmente los siguientes comandos harán lo correcto:

```
cd /home/nuevo-nombre-de-usuario
chown -R nombre-de-usuario:group .
chmod -R 755 .
```

   - Asigne una contraseña con el comando *passwd*
   - Crea un script python que cree un usuario, para ello debe recibir el nombre de usuario y nombre completo por parámetros, por defecto se pone uid y gid a 2000. Mejorar el programa para que:
   - Da un error si se intenta dar de alta un usuario que ya existe
   - Al ir dando de alta a distintos usuarios vaya incrementando automáticamente el uid y el gid a partir de 2000

1.3 Introducción a vagrant

Vagrant es una aplicación libre desarrollada en ruby que nos permite crear y personalizar entornos de desarrollo livianos, reproducibles y portables. Vagrant nos permite automatizar la creación y gestión de máquinas virtuales. Las máquinas virtuales creadas por vagrant se pueden ejecutar con distintos gestores de máquinas virtuales (oficialmente VirtualBox, VMWare e Hyper-V), en nuestro ejemplo vamos a usar máquinas virtuales en VirtualBox.

El objetivo principal de vagrant es aproximar los entornos de desarrollo y producción, de esta manera el desarrollador tiene a su disposición una manera muy sencilla de desplegar una infraestructura similar a la que se va a tener en entornos de producción. A los administradores de sistemas les facilita la creación de infraestructuras de prueba y desarrollo.

Presentación: Vagrant y ansible. Una combinación explosiva (1ª parte)

1.3.1 Práctica con vagrant

- **Práctica 1: Instalación de vagrant**

Instalar virtualbox y vagrant:

```
root@maquina:~$ apt-get install virtualbox
root@maquina:~$ wget https://releases.hashicorp.com/vagrant/2.0.0/vagrant_2.0.0_x86_
↪64.deb
root@maquina:~$ dpkg -i vagrant_2.0.0_x86_64.deb
```

- **Práctica 2: Instalación de un «box» debian/stretch**

Nos descargamos desde el repositorio oficial el box de Debian stretch de 64 bits, esto lo hacemos un usuario sin privilegios:

```
usuario@maquina:~$ vagrant box add debian/stretch64
```

Si el box lo tenemos en la *nas* de nuestro instituto:

```
usuario@maquina:~$ vagrant box add debian/stretch64 http://nas.gonzalonazareno.org/...
```

Nota: Es importante fijarnos que lo estamos haciendo con usuarios sin privilegios. Cada usuario tendrás sus box propios.

Puedo ver la lista de boxes que tengo instalada en mi usuario ejecutando la siguiente instrucción:

```
usuario@maquina:~$ vagrant box list
```

- **Práctica 3: Creación de una máquina virtual**

1. Nos creamos un directorio y dentro vamos a crear el fichero Vagrantfile, podemos crear uno vacío con la instrucción:

   ```
   usuario@maquina:~/vagrant$ vagrant init
   ```

2. Modificamos el fichero Vagrantfile y los dejamos de la siguiente manera:

   ```
   # -*- mode: ruby -*-
   # vi: set ft=ruby :
   Vagrant.configure("2") do |config|
               config.vm.box = "debian/stretch64"
               config.vm.hostname = "mimaquina"
               config.vm.network :public_network,:bridge=>"eth0"
   end
   ```

3. Iniciamos la máquina:

   ```
   usuario@maquina:~/vagrant$ vagrant up
   ```

4. Para acceder a la instancia:

   ```
   usuario@maquina:~/vagrant$ vagrant ssh default
   ```

5. Suspender, apagar o destruir:

   ```
   usuario@maquina:~/vagrant$ vagrant suspend
   usuario@maquina:~/vagrant$ vagrant halt
   usuario@maquina:~/vagrant$ vagrant destroy
   ```

Advertencia:

1. Entra en virtualbox y comprueba las características de la máquina que se ha creado.
2. ¿Qué usuario tiene creado por defecto el sistema?¿Cómo se ejecutan instrucciones de superusuario?
3. ¿Cuantas tarjetas de red tiene?¿Para qué sirve la eth0?
4. Investiga el funcionamiento de la instrucción `vagrant ssh`. ¿Por que interfaz se conecta? ¿Qué certificado se utiliza para acceder?

- **Práctica 4: Creación de varias máquinas virtuales**

En esta ocasión vamos a crear otro directorio y dentro un fichero Vagrantfile con el siguiente contenido:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|

  config.vm.define :nodo1 do |nodo1|
    nodo1.vm.box = "debian/stretch64"
    nodo1.vm.hostname = "nodo1"
    nodo1.vm.network :private_network, ip: "10.1.1.101"
  end
  config.vm.define :nodo2 do |nodo2|
    nodo2.vm.box = "debian/stretch64"
    nodo2.vm.hostname = "nodo2"
    nodo2.vm.network :public_network,:bridge=>"eth0"
    nodo2.vm.network :private_network, ip: "10.1.1.102"
  end
end
```

Cuando iniciemos el escenario veremos que hemos creado dos máquinas virtuales: nodo1 y nodo2. nodo1 tendrá una red interna con ip 10.1.1.101, y nodo2 tendrá una interfaz de red «modo puente» y una interfaz de red del tipo red interna con ip 10.1.1.102.

Si accedemos por ssh a nodo1 podremos hacer ping a nodo2.

- **Práctica 5: Añadir un dico duro adicional y modificar la RAM a una máquina virtual**

Por últimos vamos a crear un nuevo Vagranfile en un nuevo directorio con este contenido:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|

  config.vm.define :nodo1 do |nodo1|
    nodo1.vm.box = "debian/jessie64"
    nodo1.vm.hostname = "nodo1"
    nodo1.vm.network :private_network, ip: "10.1.1.101"
    nodo1.vm.provider :virtualbox do |v|
                    v.customize ["modifyvm", :id, "--memory", 768]
            end

  end

  disco = '.vagrant/midisco.vdi'
```

(continues on next page)

(proviene de la página anterior)

```
  config.vm.define :nodo2 do |nodo2|
    nodo2.vm.box = "debian/jessie64"
    nodo2.vm.hostname = "nodo2"
    nodo2.vm.network :public_network,:bridge=>"eth0"
    nodo2.vm.network :private_network, ip: "10.1.1.102"
    nodo2.vm.provider :virtualbox do |v|
                    v.customize ["createhd", "--filename", disco, "--size", 1024]
                    v.customize ["storageattach", :id, "--storagectl", "SATA␣
↪Controller",
                     "--port", 1, "--device", 0, "--type", "hdd",
                     "--medium", disco]
                    end
    end
end
```

Como podemos ver al `nodo1` le hemos modificado el tamaño de la memoria RAM y en el `nodo2` hemos añadido un disco duro de 1GB. Para que estos cambios tengan efecto debes ejecutar la instrucción:

```
usuario@maquina:~/vagrant$ vagrant reload
```

Para terminar, indicar que tenemos más parámetros de configuración que nos permiten configurar otros aspectos de la máquina virtual. Puedes encontrar más información en la documentación oficial de vagrant

1.3.2 Enlaces interesantes

- Página oficial de Vagrant
- Gestionando máquinas virtuales con Vagrant
- Boxes oficiales para Vagrant

CAPÍTULO 2

Servidor DHCP

El protocolo de configuración dinámica de host (**DHCP, Dynamic Host Configuration Protocol**),es un estándar TCP/IP que simplifica la administración de la configuración IP haciéndola automática. El servidor DHCP recibe peticiones de clientes solicitando una configuración de red IP. Responde proporcionando los parámetros que permitan a los clientes autoconfigurarse. Los clientes hay que configurarlo seleccionando la opción *"Obtener dirección IP automáticamente"*. El servidor DHCP proporciona la configuración red, entre los datos que ofrece:

- Dirección IP
- Máscara de red
- Dirección del servidor DNS
- Nombre DNS
- Puerta de enlace de la dirección IP
- Dirección de Publicación Masiva (broadcast address)
- MTU (Unidad de Transferencia Máxima según siglas en inglés) para la interfaz
- Servidores NTP (Protocolo de Tiempo de Red)
- Servidor SMTP
- Servidor TFTP

Al trabajar con el servidor DHCP tenemos que conocer los siguientes conceptos:

- **Ámbito servidor DHCP**: Agrupamiento administrativo de equipos o clientes de una subred que utilizan el servicio DHCP.
- **Rango servidor DHCP**: Grupo de direcciones IP en una subred que el servidor puede conceder a los clientes
- **Concesión o alquiler de direcciones**: Período de tiempo que los servidores DHCP especifican, durante el cual un equipo cliente puede utilizar una dirección IP.
- **Reserva de direcciones IP**: Direcciones IP que se asignan siempre a las mismos equipos. Es similar a configurar una dirección IP estática pero de forma automática desde el servidor DHCP, la forma de hacerlo es asociar direcciones MAC a direcciones IP.

- **Exclusiones**: Conjunto de direcciones IP pertenecientes al rango que no se van a asignar.

Conocer los conceptos básicos sobre configuración dinámica con IPv6:

- **SLAAC** (Stateless Address Autoconfiguration) es un método en el cual un dispositivo puede obtener una dirección IPv6 de unidifusión global sin los servicios de un servidor de DHCPv6.
- La opción de **DHCPv6** sin estado informa al cliente que utilice la información del mensaje RA para el direccionamiento, pero que hay más parámetros de configuración disponibles de un servidor de DHCPv6.
- **DHCPv6** con estado: Toda la información de direccionamiento y de configuración debe obtenerse de un servidor de DHCPv6 con estado.
- La **delegación de prefijo DHCPv6** provee de un método automatizado para que un cliente DHCPv6 solicite un prefijo IPv6 desde un servidor DHCPv6.

Indice

2.1 Enlaces interesantes

- Wikipedia - APIPA
- Internet Systems Consortium - dhcp
- Artículo: How to Install the DHCP Server on Ubuntu 12.04LTS
- Servidor dhcp en Windows 2008 Server

2.1.1 DHCPv6

- Una de IPv6, por favor (1)
- RA y DCHVPv6 Cisco Academy
- DHCPv6: Stateful Address Autoconfiguration on Linux (Part 1: radvd)
- DHCPv6 Based IPv6 Access Services
- resolvconf: Gestionando /etc/resolv.conf
- IPv6 and DNS

2.2 ¿Cómo funciona el servidor DHCP

El protocolo de configuración dinámica de host (Dynamic Host Configuration Protocol – DHCP) es una extensión de protocolo BOOTP que da más flexibilidad al administrar las direcciones IP. Este protocolo puede usarse para configurar dinámicamente los parámetros esenciales TCP/IP de los hosts (estaciones de trabajo y servidores) de una red. El protocolo DHCP tiene dos elementos:

- Un mecanismo para asignar direcciones IP y otros parámetros TCP/IP.
- Un protocolo para negociar y transmitir información específica del host.

El host TCP/IP que solicita la información de configuración TCP/IP se denomina cliente DHCP y el host que provee dicha información se llama servidor DHCP. El DHCP se describe en la norma **RFC 2131** –Protocolo de configuración dinámica de host–. A continuación, presentamos la operación del DHCP.

2.2.1 Administración de direcciones con el DHCP

El protocolo DHCP usa los siguientes 3 métodos para asignar las direcciones IP:

a) Asignación manual El administrador de red pone manualmente la dirección IP del cliente DHCP en el servidor DHCP. El DHCP se usa para dar al cliente DHCP el valor de esta dirección IP configurada manualmente.

b) Asignación automática No se requiere asignar manualmente direcciones IP. El servidor DHCP asigna al cliente DHCP, en el primer contacto, una dirección IP permanente que no podrá reutilizar ningún otro cliente DHCP.

c) Asignación dinámica El DHCP asigna una dirección IP al cliente DHCP por un tiempo determinado. Después que expire este lapso, se revoca la dirección IP y el cliente DHCP tiene que devolverla. Si el cliente aún necesita una dirección IP para efectuar sus operaciones, deberá solicitarla nuevamente.

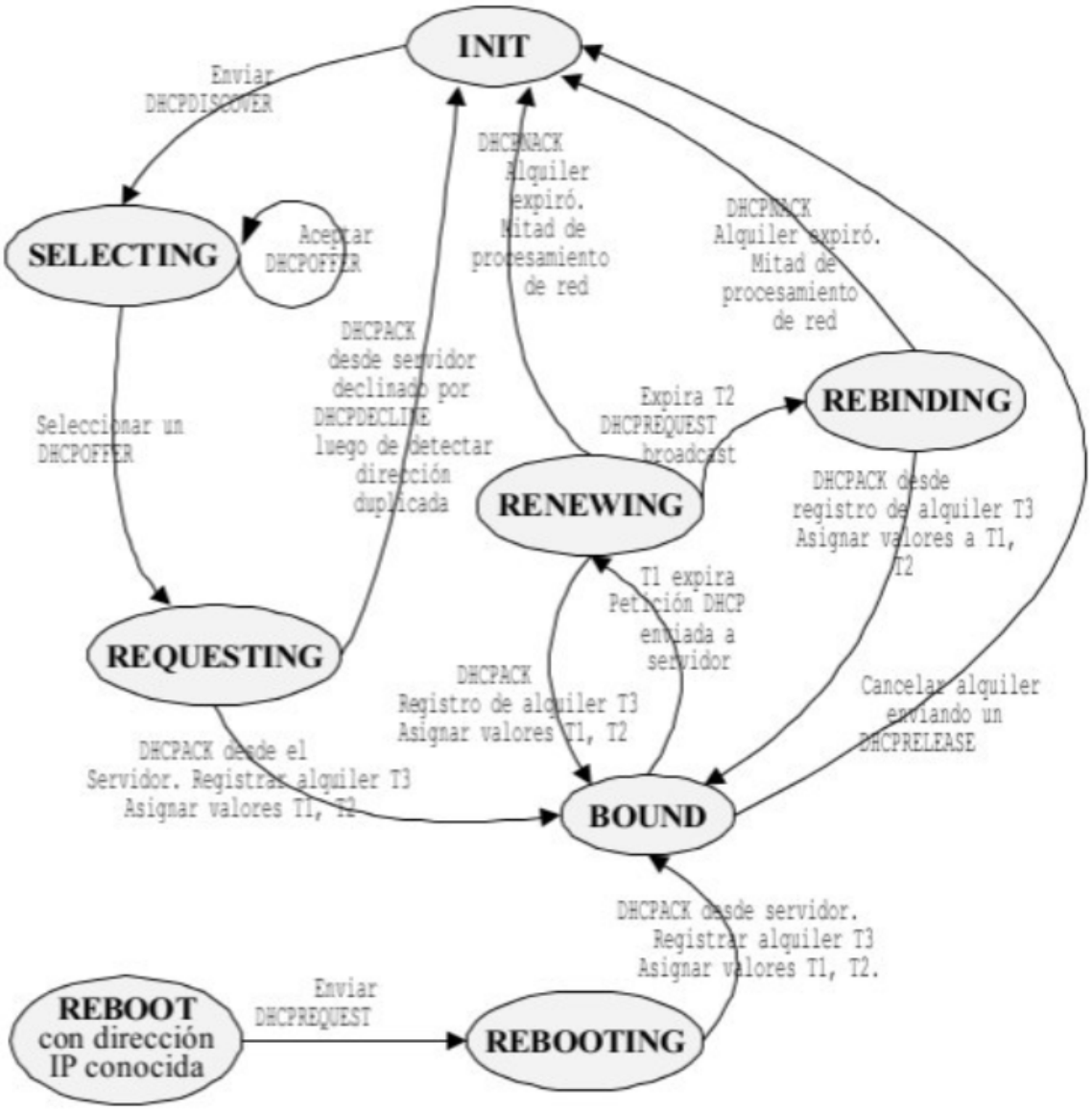
Este protocolo permite la reutilización automática de una dirección IP. Si un cliente DHCP ya no necesita una dirección IP, como en el caso de un ordenador que apagamos, éste libera su dirección y la entrega al servidor DHCP. Éste puede reasignar dicha dirección a otro cliente que la pida.

El método de asignación dinámica es muy útil para clientes DHCP que necesitan una dirección IP para una conexión temporal a la red. Por ejemplo, consideremos una situación en que 300 usuarios tengan ordenadores portátiles conectadas a una red y ésta les ha asignado direcciones clase C. Este tipo de dirección permite a la red tener hasta 253 nodos (255 – 2 direcciones especiales = 253). Debido a que los ordenadores que se conectan a una red usando el TCP/IP requieren tener una dirección única IP, entonces las 300 ordenadores no podrían operar simultáneamente. Sin embargo, si sólo hay 200 conexiones físicas a la red se puede buscar una dirección de clase C mediante la reutilización de direcciones IP no usadas. Usando el DHCP, en su método de asignación dinámica de direcciones IP, es posible reutilizar direcciones IP.

Además la asignación dinámica de direcciones IP es un buen método para asignar direcciones IP a ordenadores que van a ser conectados por primera vez y en una red donde escasean las direcciones IP. Si los ordenadores antiguos se retiran, sus direcciones IP pueden ser reutilizadas o reasignadas inmediatamente. Sin importar cuál método se elija, aún puede configurarse los parámetros IP de una sola vez desde un servidor central, en lugar de repetir la configuración TCP/IP para cada ordenador.

2.2.2 Proceso de configuración de los clientes

Una vez que un cliente DHCP ha contactado con un servidor DHCP, a través de varios estados internos, negocia el uso y la duración de su dirección IP. La forma de adquisición de la dirección IP por el cliente DHCP se explica mejor en términos de un diagrama de transición de estados (llamado también máquina de estado finito) . La figura presenta este diagrama de transición de estados que explica la interacción entre el cliente y el servidor DHCP.



Descubrimiento de un servidor DHCP (SELECTING)

Cuando se inicializa el cliente DHCP, éste comienza en el estado de inicialización INIT. El cliente DHCP desconoce sus parámetros IP y por eso envía un broadcast DHCPDISCOVER. El mensaje DHCPDISCOVER se encapsula en un paquete UDP. Se coloca el número 67 con puerta de destino UDP, el mismo utilizado por el servidor BOOTP, debido a que el protocolo DHCP es una extensión de este protocolo. El mensaje DHCPDISCOVER usa la dirección IP de broadcast de valor 255. 255. 255. 255. Si no existe un servidor DHCP en la red local, el router IP debe tener un agente DHCP relay que soporte la retransmisión de esta petición hacia las otras subredes. El agente DHCP relay se describe en la norma RFC 1542.

Antes de enviar el mensaje broadcast DHCPDISCOVER, el cliente DHCP espera por un tiempo aleatorio –entre 1 a 10 segundos– para evitar una colisión con otro cliente DHCP, como en el caso que todos los clientes DHCP se inicial icen al mismo tiempo al recibir todos energía a la vez (como una pérdida o interrupción de la electricidad).

Aceptación de la asignación recibida (REQUESTING)

Después de enviar el mensaje broadcast DHCPDISCOVER, el cliente DHCP ingresa al estado SELECTING, donde

recibe los mensajes DHCPOFFER de los servidores DHCP configurados para atenderlo. El tiempo que el cliente DHCP esperará por los mensajes DHCPOFFER depende de la implementación. Si el cliente DHCP recibe varias respuestas DHCPOFFER, elegirá una. En reacción, el cliente DHCP enviará un mensaje DHCPREQUEST para elegir un servidor DHCP, el que contestará con un DHCPACK.

Como opción, el cliente DHCP controla la dirección IP enviada en el DHCPACK para verificar si está o no está en uso. En una red con broadcast, el cliente DHCP envía una petición ARP con la dirección IP sugerida para verificar que no esté duplicada. En caso de estarlo, el DHCPACK proveniente del servidor se ignora y se envía un DHCPDECLINE, con lo cual el cliente DHCP ingresa en estado INIT y vuelve a pedir una dirección IP válida que no esté en uso.

Cuando la petición ARP se difunde sobre la red, el cliente usa su propia dirección de hardware en el campo de dirección fuente de hardware del ARP, pero coloca el valor de 0 en el campo de dirección fuente IP. Esta dirección de valor 0 se utiliza en lugar de la dirección IP sugerida, para no confundir a las memorias caché ARP de otros hosts.

Duración de la concesión (BOUND)

Cuando se acepta el DHCPACK proveniente del servidor DHCP, se colocan tres valores de temporización y el cliente DHCP se mueve al estado BOUND (asociado) .

```
* T1 es el temporizador de renovación de alquiler.
* T2 es el temporizador de reenganche.
* T3 es la duración del alquiler.
```

El DHCPACK siempre trae consigo el valor de T3. Los valores de T1 y T2 se configuran en el servidor DHCP; de no ser así, se usan los valores por defecto siguientes:

```
* T1 = 0,5 x T3.
* T2 = 0,875 x T3.
```

El tiempo actual en que los temporizadores expiran se calcula añadiendo el valor del temporizador al tiempo en que se envió el mensaje DHCPREQUEST, el cual generó la respuesta DHCPACK.

Si este tiempo es T0, entonces los valores de expiración se calculan así:

```
* Expiración de T1 = T0 + T1
* Expiración de T2 = T0 + T2
* Expiración de T3 = T0 + T3
```

La RFC 2131 recomienda que se debe añadir un factor a T1 y T2 para evitar que varios clientes DHCP expiren sus temporizadores al mismo tiempo.

Renovación de la concesión (RENEWING)

Después de la expiración del temporizador T1, el cliente DHCP se mueve del estado BOUND al estado RENEWING (renovación) . En este último estado se debe negociar un nuevo alquiler para la dirección IP designada, entre el cliente DHCP y el servidor DHCP que originalmente le asignó la dirección IP. Si el servidor DHCP original, por algún motivo, no renueva el alquiler, le enviará un mensaje DHCPNACK y el cliente DHCP se moverá al estado INIT y intentará obtener una nueva dirección IP. En el caso contrario, si el servidor DHCP original envía un mensaje DHCPACK, éste contendrá la duración del nuevo alquiler. Entonces, el cliente DHCP coloca los valores de sus temporizadores y se moverá al estado BOUND.

Estado de reenganche (RENEWING)

Si el temporizador T2 (tiempo de reenganche) expira mientras el cliente DHCP está esperando en el estado RENEWING una respuesta sea DHCPACK o DHCPNACK proveniente del servidor DHCP original, el cliente DHCP se moverá al estado REBINDING. El servidor original DHCP podría no haber respondido porque estaría apagado o porque el enlace con la red habría caído. Nótese en las ecuaciones previas que T2 es mayor que T1, de modo que el cliente DHCP espera que el servidor original DHCP renueve el alquiler por un tiempo igual a T2 – T1.

Extensión de la concesión

Al expirar el temporizador T2 (tiempo de reenganche), el cliente DHCP enviará un DHCPREQUEST a la red para contactar con cualquier servidor DHCP para extender el alquiler, con lo cual pasará al estado REBINDING. El cliente DHCP envía este mensaje broadcast DHCPREQUEST porque presume que, luego de haber esperado T2 – T1 segundos en el estado RENEWING, el servidor DHCP original no está disponible, por lo cual tratará de contactar con otro servidor DHCP para que le responda. Si un servidor DHCP responde con un DHCPACK, el cliente DHCP renueva su alquiler (T3), coloca los temporizadores T1 y T2 y retorna al estado BOUND. Si no hay servidor DHCP disponible para renovar alquiler luego de expirar el temporizador T3, el alquiler cesa y el cliente DHCP pasa al estado INIT. Nótese que el cliente DHCP intentó renovar el alquiler primero con el servidor original y luego con cualquier otro servidor en la red.

Expiración de la concesión

Al acabar el alquiler (T3 expira), el cliente DHCP debe devolver su dirección IP y cesar toda acción con dicha dirección IP en la red. El cliente DHCP no siempre tiene que esperar la expiración del alquiler para terminar el uso de una dirección IP. Éste puede renunciar voluntariamente a una dirección IP, cancelando su alquiler. Por ejemplo, el usuario de un computador portátil podría conectarse a la red para una actividad particular. El servidor DHCP de la red podría colocar la dirección del alquiler por una hora. Suponiendo que el usuario acabe su tarea en 30 minutos, entonces se desconectará de la red al cabo de dicho lapso. Cuando el usuario se libera armoniosamente, el cliente DHCP enviará un mensaje DHCPRELEASE al servidor DHCP para cancelar el alquiler. La dirección IP ahora estará disponible.

Si los clientes DHCP operan en ordenadores que tienen disco duro, la dirección IP asignada puede ser almacenada en este dispositivo y, cuando la computadora reinicie sus operaciones, puede hacer una nueva petición usando esta dirección IP.

2.3 Cuestionario DHCP

1. ¿Qué es un servidor DHCP?
2. ¿Cómo hay que configurar un cliente DHCP?
3. Ventajas de usar un servidor DHCP
4. Enumera los distintos parámetros que un servidor DHCP puede conceder a un cliente
5. Definición de:
   - Ámbito servidor DHCP
   - Rango servidor DHCP
   - Concesión o alquiler de direcciones
   - Reserva de direcciones IP
   - Exclusión de direcciones IP
6. ¿Qué es APIPA?
7. ¿Por qué el mensaje DHCPDISCOVER es del tipo broadcast?
8. Desde el estado INIT al estado BOUND. ¿Qué mensajes se transmiten desde el cliente al servidor?
9. ¿Qué significa el mensaje DHCPDECLINE por parte del cliente?
10. Una vez que se ha aceptado una asignación (DHCPACK), explica los siguientes tiempos:
    - T1: tiempo de renovación de alquiler
    - T2: tiempo de reenganche
    - T3: tiempo de concesión del alquiler
11. ¿Qué es la renovación (RENEWING) de alquiler?

12. ¿Qué es el estado de reenganche (REBINDING)?
13. ¿Qué ocurre cuando termina el tiempo de concesión del alquiler (T3)?
14. ¿Qué ocurre cuando un cliente manda un DHCPRELEASE?
15. ¿Qué ocurre cuando un cliente que ya tiene asignación se reinicia?
16. Los clientes toman una configuración, y a continuación apagamos el servidor dhcp. ¿qué ocurre con el cliente windows? ¿Y con el cliente linux?
17. Los clientes toman una configuración, y a continuación cambiamos la configuración del servidor dhcp (por ejemplo el rango). ¿qué ocurriría con un cliente windows? ¿Y con el cliente linux?
18. ¿Qué instrucciones en Windows y en Linux nos permiten?:
    - Para renovar una dirección IP y una nueva configuración de red
    - Para liberar la dirección IP
19. ¿Qué es la lista de concesiones? ¿En qué fichero se guarda en Linux?
20. En el servidor isc-dhcp en linux. ¿Qué indican los siguientes parámetros?:
    - max-lease-time
    - default-lease-time
21. En el cliente, ¿qué se guarda en los ficheros `dhclient-???.lease`?

2.4 Ejercicio: Instalación y configuración del servidor dhcp en linux

Después de leer la documentación, instala el servidor dhcp. Recuerda que al inicializar el servicio nos dará un error, esto es debido a que no hemos configurado el servidor.

2.4.1 Instalación del servidor isc-dhcp-server

Para instalar nuestro servidor dhcp ejecutamos:

```
apt-get install isc-dhcp-server
```

Nota: Cuando instalamos el servidor por primera se produce un error, ya que no está configurado. Puedes ver los errores producidos por el servidor en el fichero /var/log/syslog

2.4.2 Configuración del servidor isc-dhcp-server

Lo primero que tenemos que hacer es configurar el interfaz de red por el que va a trabajar el servidor dhcp, para ello editamos el siguiente fichero `/etc/default/isc-dhcp-server`.

Donde configuramos el parámetro interfaces, por ejemplo:

```
INTERFACES="eth1"
```

El fichero principal de configuración de DHCP es `/etc/dhcp/dhcpd.conf`.

El fichero de configuración está dividido en dos partes:

- Parte principal (valores por defecto): especifica los parámetros generales que definen la concesión y los parámetros adicionales que se proporcionarán al cliente.
- Secciones (concretan a la principal)
  - Subnet: Especifican rangos de direcciones IPs que serán cedidas a los clientes que lo soliciten.
  - Host: Especificaciones concretas de equipos.

En la parte principal podemos configurar los siguientes parámetros, que más tarde podremos reescribir en las distintas secciones:

- `max-lease-time`: Tiempo de la concesión de la dirección IP
- `default-lease-time`: Tiempo de renovación de la concesión
- `option routers`: Indicamos la dirección red de la puerta de enlace que se utiliza para salir a internet.
- `option domain-name-server`: Se pone las direcciones IP de los servidores DNS que va a utilizar el cliente.
- `option domain-name`: Nombre del dominio que se manda al cliente.
- `option subnetmask`: Subred enviada a los clientes.
- `option broadcast-address`: Dirección de difusión de la red.

Al indicar una sección subnet tenemos que indicar la dirección de la red y la mascara de red y entre llaves podemos poner los siguientes parámetros:

- `range`: Indicamos el rango de direcciones IP que vamos a asignar.
- Algunos de los parámetros que hemos explicado en la sección principal.

Ejemplo de configuración de la sección subnet puede ser:

```
subnet 192.168.0.0 netmask 255.255.255.0 {
  range 192.168.0.60 192.168.0.90;
  option routers 192.168.0.254;
  option domain-name-server 80.58.0.33, 80.58.32.9;
}
```

Reinciciamos el servidor dhcp:

```
service isc-dhcp-server restart
```

Sólo falta configurar los clientes para que tomen la configuración de red de forma dinámica.

Nota: En Windows la instrucción `ipconfig /release` libera la concesión, la instrucción `ipconfig /renew` la renueva. En linux el comando para liberar la concesión es `dhclient -r` y el que nos permite renovarla será `dhclient`.

Advertencia: Ejercicios

1. Configura el servidor dhcp con las siguientes características

- Rango de direcciones a repartir: 192.168.0.100 - 192.168.0.110
- Máscara de red: 255.255.255.0
- Duración de la concesión: 1 hora

- Puerta de enlace: 192.168.0.1
- Servidores DNS: 8.8.8.8, 8.8.4.4

2. Configura los clientes para obtener direccionamiento dinámico. Comprueba las configuraciones de red que han tomado los clientes. Visualiza el fichero del servidor donde se guarda las configuraciones asignadas.

2.4.3 Creando reservas

Veamos la sección host, en ella configuramos un host para reservar una dirección IP para él.

En una sección host debemos poner el nombre que identifica al host y los siguientes parámetros:

- `hardware ethernet`: Es la dirección MAC de la tarjeta de red del host.
- `fixed-address`: La dirección IP que le vamos a asignar.
- Podemos usar también las opciones ya explicadas en la sección principal.

Advertencia: Ejercicios

1. Crea en el servidor dhcp una sección HOST para conceder a un cliente una dirección IP determinada (por ejemplo la 192.168.0.105)
2. Obtén una nueva dirección IP en el cliente y comprueba que es la que has asignado por medio de la sección host.

Advertencia: Realiza las siguientes comprobaciones

Vamos a comprobar que ocurre con la configuración de los clientes en determinadas circunstacia, para ello vamos a poner un tiempo de conseción muy bajo.

1. Los clientes toman una configuración, y a continuación apagamos el servidor dhcp. ¿qué ocurre con el cliente windows? ¿Y con el cliente linux?
2. Los clientes toman una configuración, y a continuación cambiamos la configuración del servidor dhcp (por ejemplo el rango). ¿qué ocurre con el cliente windows? ¿Y con el cliente linux?

CAPÍTULO 3

Servidor Web

En este bloque del módulo vamos a estudiar el servidor Web, programa software que utilizando el protocolo HTTP, es capaz de procesor en el servidor la peticiones HTTP y generar las respuestas adecuadas. En este bloque vamos a estudiar distintos aspectos del servicio web.

- El protocolo HTTP: tipos de peticiones, tipos de respuestas, cabeceras, autentificación, control de acceso, etc.
- Vamos a usar el servidor Apache 2.4 y el servidor nginx:
  - Configuración básica
  - Virtual Hosting
  - Mapeo de URL
  - Autentificación y control de acceso
  - Módulos
  - Módulos de multiprocesamiento
  - Ejecución de scripts en el servidor: PHP, Python,...

Indice

3.1 Enlaces interesantes

Protocolo HTTP

- Introducción al protocolo HTTP
- Protocolo HTTP (Univ. Valencia) Vídeo 1 Vídeo 1)
- Protocolo HTTP (Univ. Valencia) Vídeo 2
- Qué son los MIME Types
- Lista de cabeceras HTTP 1.1

Servidor web Apache

- Documentación oficial servidor web Apache 2.2
- Documentación oficial servidor web Apache 2.4
- Updating Virtual Host Settings from Apache 2.2 to Apache 2.4
- How To Migrate your Apache Configuration from 2.2 to 2.4 Syntax

Módulos en apache

- Utilización de módulos en Apache
- Cómo crear URLS amigables con htaccess

Servidor nginx

- Documentación oficial de nginx

3.2 Ejercicio: Hacer peticiones HTTP: GET, HEAD y POST

1. Utilizando el comando de linux HEAD visualiza la información de la cabeceras de los URL:

```
http://dit.gonzalonazareno.org
http://informatica.gonzalonazareno.org/proyectos/index.html
http://josedom24.github.io/img/yo1.jpg
```

   Identifica todos los parámetros que puedas.

   Utiliza el plugin de firefox **HttpFox** para identificar las cabeceras de las peticiones y de las respuestas.

1. Utilizando el método GET obtén el contenido de la página:

```
http://dit.gonzalonazareno.org/moodle/index.php
http://dit http://www.debian.org/index.html
```

   Observa con HttpFox cuantas peticiones se realizan al acceder a estas páginas.

1. Envío de información al servidor, comprueba como se manda información al servidor mediante el método GET en la URL:

```
http://playerone.josedomingo.org/ejget.php?valor=hola
http://dit.gonzalonazareno.org/moodle/course/view.php?id=4
```

   Usando el comando GET manda tu nombre a la página:

```
http://playerone.josedomingo.org/ejget.php
```

   Usando el comando POST (que envia el contenido en el cuerpo) manda tu nombre a la página:

```
http://playerone.josedomingo.org/ejpost.php
```

3.3 Ejercicio: Instalación y configuración básica de Apache

3.3.1 Instalación de Apache 2.4

1. Instala el servidor web Apache:

apt-get install apache2

Para controlar el servicio apache2 podemos usar (para más información):

```
apache2ctl [-k start|restart|graceful|graceful-stop|stop]
```

1. ¿Qué es la opción graceful?
2. Comprueba la directiva donde indicamos el puerto de escucha del servidor. Modifica el puerto de escucha para que sea el 8080. Comprueba el acceso al servidor desde un navegador.

3.3.2 Estructura de los ficheros de configuración

El fichero principal de configuración de Apache2 es `/etc/apache2/apache2.conf`. En ese fichero se incluyen los ficheros que forman parte de la configuración de Apache2:

```
...
IncludeOptional mods-enabled/*.load
IncludeOptional mods-enabled/*.conf
...
Include ports.conf
...
IncludeOptional conf-enabled/*.conf
IncludeOptional sites-enabled/*.conf
```

Por defecto se indican las opciones de configuración del directorio `/var/www` y de todos sus subdirectorios, por lo tanto los `DocumentRoot` de los virtualhost que se crean deben ser subdirectorios del este directorio:

```
<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

Podemos indicar como directorio raíz de nuestros virtualhost otro directorio (tenemos que descomentar):

```
#<Directory /srv/>
#    Options Indexes FollowSymLinks
#    AllowOverride None
#    Require all granted
#</Directory>
```

3.3.3 Algunas directivas

- IfDefine.Las directivas de configuración de apache2 se pueden aplicar si está definido un determinado parámetro
- IfModule. Podemos aplicar determinadas directivas si hay cargado un determinado módulo.
- LoadModule: Nos permite cargar dinámicamente los módulos.
- Include nos permite añadir ficheros de configuración a la configuración general de apache2.

Podemos aplicar directivas a partes concretas de nuestro servidor web, para ello estudia las siguientes directivas (Para aprender más lee Secciones de Configuración):

- Directory
- DirectoryMatch

- Files
- FilesMatch
- Location
- LocationMatch
- VirtualHost

Directivas de identificación del servidor:

- ServerName
- ServerAdmin
- ServerTokens

Directivas de localización de ficheros

- DocumentRoot
- ErrorLog
- CustomLog
- LockFile
- PidFile
- ServerRoot
- AccessFileName

Directivas de control de la conexión

- Timeout
- KeepAlive (Más información)
- MaxKeepAliveRequests
- KeepAliveTimeout

Otras directivas

- User
- Group
- DefaultType
- LogLevel
- LogFormat

3.4 Ejercicio: VirtualHosting con Apache

3.4.1 Introducción al VirtualHosting

El término Hosting Virtual se refiere a hacer funcionar más de un sitio web (tales como `www.company1.com` y `www.company2.com`) en una sola máquina. Los sitios web virtuales pueden estar «basados en direcciones IP», lo que significa que cada sitio web tiene una dirección IP diferente, o «basados en nombres diferentes», lo que significa que con una sola dirección IP están funcionando sitios web con diferentes nombres (de dominio). Apache fue uno de los primeros servidores web en soportar hosting virtual basado en direcciones IP.

El servidor web Apache2 se instala por defecto con un host virtual. La configuración de este sitio la podemos encontrar en:

```
/etc/apache2/sites-available/000-default.conf
```

Cuyo contenido podemos ver:

```
<VirtualHost *:80>
        #ServerName www.example.com
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Donde encontramos los siguientes parámetros:

- ServerName
- ServerAdmin
- DocumentRoot
- ErrorLog
- CustomLog

Y por defecto este sitio virtual está habilitado, por lo que podemos comprobar que existe un enlace simbólico a este fichero en el directorio `/etc/apache2/sites-enables`:

```
lrwxrwxrwx 1 root root   35 Oct  3 15:24 000-default.conf -> ../sites-available/000-
↪default.conf
```

Podemos habilitar o deshabilitar nuestros host virtuales utilizando los siguientes comandos:

```
a2ensite
a2dissite
```

En el fichero de configuración general `/etc/apache2/apache2.conf` nos encontramos las opciones de configuración del directorio padre del indicado en la directiva `DocumentRoot` (suponemos que todos los host virtuales van a estar guardados en subdirectorios de este directorio):

```
...
<Directory /var/www/>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
...
```

3.4.2 Configuración de VirtualHosting

El objetivo de esta práctica es la puesta en marcha de dos sitios web utilizando el mismo servidor web apache. Hay que tener en cuenta lo siguiente:

- Cada sitio web tendrá nombres distintos.
- Cada sitio web compartirán la misma dirección IP y el mismo puerto (80).

Queremos construir en nuestro servidor web apache dos sitios web con las siguientes características:

- El nombre de dominio del primero será `www.iesgn.org`, su directorio base será `/var/www/iesgn` y contendrá una página llamada `index.html`, donde sólo se verá una bienvenida a la página del Instituto Gonzalo Nazareno.
- En el segundo sitio vamos a crear una página donde se pondrán noticias por parte de los departamento, el nombre de este sitio será `www.departamentosgn.org`, y su directorio base será `/var/www/departamentos`. En este sitio sólo tendremos una página inicial `index.html`, dando la bienvenida a la página de los departamentos del instituto.

Para conseguir estos dos sitios virtuales debes seguir los siguientes pasos:

1. Los ficheros de configuración de los sitios webs se encuentran en el directorio `/etc/apache2/sites-available`, por defecto hay dos ficheros, uno se llama `000-default.conf` que es la configuración del sitio web por defecto. Necesitamos tener dos ficheros para realizar la configuración de los dos sitios virtuales, para ello vamos a copiar el fichero `000-default.conf`:

   cd /etc/apache2/sites-available cp 000-default.conf iesgn.conf cp 000-default.conf departamentos.conf

   De esta manera tendremos un fichero llamado `iesgn.conf` para realizar la configuración del sitio web `www.iesgn.org`, y otro llamado `departamentos.conf` para el sitio web `www.departamentosgn.org`.

2. Modificamos los ficheros `iesgn.conf` y `departamentos.conf`, para indicar el nombre que vamos ausar para acceder al host virtual (`ServerName`) y el directorio detrabajo (`DocumentRoot`).

3. No es suficiente crear los ficheros de configuración de cada sitio web, es necesario crear un enlace simbólico a estos ficheros dentro del directorio `/etc/apache2/sites-enabled`, para ello:

   ```
   a2ensite iesgn
   a2ensite departamentos
   ```

   La creación de los enlaces simbólicos se puede hacer con la instrucción `a2ensite nombre_fichero_configuracion`, para deshabilitar el sitio tenemos que borrar el enlace simbólico o usar la instrucción `a2dissite nombre_fichero_configuracion`.

4. Crea los directorios y los ficheros index.html necesarios en `/var/www` y reiniciamos el servicio.

5. Para terminar lo único que tendremos que hacer es cambiar el fichero hosts en los clientes y poner dos nuevas líneas donde se haga la conversión entre los dos nombre de dominio y la dirección IP del servidor.

3.5 Ejercicio: Mapear URL a ubicaciones de un sistema de ficheros

1. Crea un nuevo host virtual que es accedido con el nombre `www.mapeo.com`, cuyo `DocumentRoot` sea /srv/mapeo.
2. Alias: Crea un alias en el host virtual del ejercicio anterior, que mi permita entrar en la URL `http://www.mapeo.com/documentos` y visualice los ficheros del `/home/usuario/Documentos`. En la sección `Directory...` pon las mismas directivas que tiene la sección `Directory` del fichero `/etc/apache2/apache2.conf`.
3. Options: Determina para que sirven las siguientes opciones de funcionamieno:
   - All
   - FollowSymLinks
   - Indexes
   - MultiViews
   - SymLinksOwnerMatch

- ExecCGI

Detemina como funciona si delante de las opciones pongo el signo + o -.

- Crea un enlace directo dentro de `/home/usuario/document` y comprueba si es posible seguirlo. Cambia las opciones del directorio para que no siga los enlaces símbolicos.
- Deshabiliata la opción de que se listen los archivos existentes en la carpeta cuando no existe un fichero definido en la directiva DirectoryIndex.
- MultiViews: Para saber más sobre el negociado de contenido. Siguiendo el ejemplo de esta 'página](http://www.howtoforge.com/using-apache2-content-negotiation-to-serve-different-languages) realiza un fichero de bienvenida en español e inglés y compruba como se visualiza.

4. Usando la directiva Redirect realiza una redirección, que permita que cuando entre a tu servidor `http://nombre_servidor`, salte a `http://nombre_servidor/web`.
5. Con la directiva `ErrorDocument` se puede crear Respuesta de error personalizadas. Todo esto se puede llevar a cabo en el fichero `/etc/apache2/conf-available/localized-error-pages.conf`. Después de leer sobre el tema realiza los siguientes ejercicios.

- Cuando no se encuentre una página (error 404) por un mensje de error.
- Crea un alias llamado `error` que corresponda a `/srv/mapeo/error`. Dentro de ese directorio crea páginas personalizadas para visualizar cuando se produzca un error 404 y cuando se tenga un forbidden (403). Configura el sistema para que se redireccione a estas páginas cuando se produce un error.
- Descomenta en el fichero `localized-error-pages.conf` las líneas adecuadas para tener los mensajes de error traducidos a los diferentes idiomas. Para que funcione tienes que hacer dos cosas:
  - Activar el módulo `include`.
  - Si quieres los mensajes en español modifica adecuadamente la directiva `LanguagePriority` del módulo `negotiation`.

3.6 Ejercicio: Control de acceso, autentificación y autorización

3.6.1 Control de acceso

El **Control de acceso** en un servidor web nos permite determinar desde donde podemos acceder a los recursos del servidor.

En **apache2.2** se utilizan las siguientes directivas: order, allow y deny. Un buen manual para que quede más claro lo puedes encontrar en este enlace. La directiva satisfy controla como el se debe comportar el servidor cuando tenemos autorizaciones de control de acceso (allow, deny,...) y tenemos autorizaciones de usuarios (require).

En **apache2.4** se utilizan las siguientes directivas: Require, RequireAll, RequireAny y RequireNone

1. Comprueba el control de acceso por defecto que tiene el virtual host por defecto (000-default).
2. Crea un escenario en Vagrant que tenga un servidor con una red publica, y una privada, un cliente conectada a la red privada. Crea un host virtual, que sólo se tenga acceso desde el cliente de la red local, y no se pueda acceder desde la anfitriona por la red pública.

3.6.2 Autentificación básica

El servidor web Apache puede acompañarse de distintos módulos para proporcionar diferentes modelos de autenticación. La primera forma que veremos es la más simple. Usamos para ello el módulo de autenticación básica que viene

instalada «de serie» con cualquier Apache: mod_auth_basic. La configuración que tenemos que añadir en el fichero de definición del Virtual Host a proteger podría ser algo así:

```
<Directory "/var/www/miweb/privado">
  AuthUserFile "/etc/apache2/claves/passwd.txt"
  AuthName "Palabra de paso"
  AuthType Basic
  Require valid-user
</Directory>
```

El método de autentificación básica se indica en la directiva AuthType.

- En `Directory` escribimos el directorio a proteger, que puede ser el raíz de nuestro Virtual Host o un directorio interior a este.
- En AuthUserFile ponemos el fichero que guardará la información de usuarios y contraseñas que debería de estar, como en este ejemplo, en un directorio que no sea visitable desde nuestro Apache. Ahora comentaremos la forma de generarlo.
- Por último, en AuthName personalizamos el mensaje que aparecerá en la ventana del navegador que nos pedirá la contraseña.
- Para controlar el control de acceso, es decir, que usuarios tienen permiso para obtener el recurso utilizamos las siguientes directivas: AuthGroupFile, Require user, Require group.

El fichero de contraseñas se genera mediante la utilidad `htpasswd`. Su sintaxis es bien sencilla. Para añadir un nuevo usuario al fichero operamos así:

```
$ htpasswd /etc/apache2/claves/passwd.txt carolina
New password:
Re-type new password:
Adding password for user carolina
```

Para crear el fichero de contraseñas con la introducción del primer usuario tenemos que añadir la opción `-c` (create) al comando anterior. Si por error la seguimos usando al incorporar nuevos usuarios borraremos todos los anteriores, así que cuidado con esto. Las contraseñas, como podemos ver a continuación, no se guardan en claro. Lo que se almacena es el resultado de aplicar una función hash:

```
josemaria:rOUetcAKYaliE
carolina:hmO6V4bM8KLdw
alberto:9RjyKKYK.xyhk
```

Para denegar el acceso a algún usuario basta con que borremos la línea correspondiente al mismo. No es necesario que le pidamos a Apache que vuelva a leer su configuración cada vez que hagamos algún cambio en este fichero de contraseñas.

La principal ventaja de este método es su sencillez. Sus inconvenientes: lo incómodo de delegar la generación de nuevos usuarios en alguien que no sea un administrador de sistemas o de hacer un front-end para que sea el propio usuario quien cambie su contraseña. Y, por supuesto, que dichas contraseñas viajan en claro a través de la red. Si queremos evitar esto último podemos crear una instancia Apache con SSL.

Cómo funciona este método de autentificación

Cuando desde el cliente intentamos acceder a una URL que esta controlada por el método de autentificación básico:

1. El servidor manda una respuesta del tipo 401 *HTTP/1.1 401 Authorization Required* con una cabecera *WWW-Authenticate* al cliente de la forma:

   ```
   WWW-Authenticate: Basic realm="Palabra de paso"
   ```

2. El navegador del cliente muestra una ventana emergente preguntando por el nombre de usuario y contraseña y cuando se rellena se manda una petición con una cabecera *Authorization*

```
Authorization: Basic am9zZTpqb3Nl
```

En realidad la información que se manda es el nombre de usuario y la contraseña en base 64, que se puede decodificar fácilmente con cualquier utilidad.

Ejercicios

1. Crea cuatro usuarios de apache: pepe, maria, juan, ana.
2. Crea dos grupos de usuarios: grupo1 (pepe,maria), grupo2 (juan,ana).
3. Crea un directorio llamado privado1 en el host virtual default, que permita el acceso a todos los usuarios.
4. Crea un directorio llamado privado2 en el host virtual default, que permita el acceso sólo a juan y a ana.
5. Crea un directorio llamado privado3 en el host virtual default, que permita el acceso sólo los usuarios del grupo1.
6. El directorio privado3 del ejercicio5 haz que sólo sea accesible desde el localhost.

3.6.3 Autentificación tipo digest

La autentificación tipo digest soluciona el problema de la transferencia de contraseñas en claro sin necesidad de usar SSL. El procedimiento, como veréis, es muy similar al tipo básico pero cambiando algunas de las directivas y usando la utilidad `htdigest` en lugar de `htpassword` para crear el fichero de contraseñas. El módulo de autenticación necesario suele venir con Apache pero no habilitado por defecto. Para activarlo usamos la utilidad `a2enmod` y, a continuación reiniciamos el servidor Apache:

```
$ a2enmod auth_digest
$ /etc/init.d/apache2 restart
```

Luego incluimos una sección como esta en el fichero de configuración de nuestro Virtual Host:

```
<Directory "/var/www/miweb/privado">
   AuthType Digest
   AuthName "dominio"
   AuthUserFile "/etc/claves/digest.txt"
   Require valid-user
</Directory>
```

Como vemos, es muy similar a la configuración necesaria en la autenticación básica. La directiva `AuthName` que en la autenticación básica se usaba para mostrar un mensaje en la ventana que pide el usuario y contraseña, ahora se usa también para identificar un nombre de dominio (realm) que debe de coincidir con el que aparezca después en el fichero de contraseñas. Dicho esto, vamos a generar dicho fichero con la utilidad htdigest:

```
$ htdigest -c /etc/claves/digest.txt dominio josemaria
Adding password for josemaria in realm dominio.
New password:
Re-type new password:
```

Al igual que ocurría con htpassword, la opción `-c` (create) sólo debemos de usarla al crear el fichero con el primer usuario. Luego añadiremos los restantes usuarios prescindiendo de ella. A continuación vemos el fichero que se genera después de añadir un segundo usuario:

```
josemaria:dominio:8d6af4e11e38ee8b51bb775895e11e0f
gemma:dominio:dbd98f4294e2a49f62a486ec070b9b8c
```

Cómo funciona este método de autentificación

Cuando desde el cliente intentamos acceder a una URL que esta controlada por el método de autentificación de tipo digest:

1. El servidor manda una respuesta del tipo 401 *HTTP/1.1 401 Authorization Required* con una cabecera *WWW-Authenticate* al cliente de la forma:

```
WWW-Authenticate: Digest realm="dominio",
                  nonce="cIIDldTpBAA=9b0ce6b8eff03f5ef8b59da45a1ddfca0bc0c485",
                  algorithm=MD5,
                  qop="auth"
```

2. El navegador del cliente muestra una ventana emergente preguntando por el nombre de usuario y contraseña y cuando se rellena se manda una petición con una cabecera *Authorization*

```
Authorization   Digest username="jose",
                realm="dominio",
                nonce="cIIDldTpBAA=9b0ce6b8eff03f5ef8b59da45a1ddfca0bc0c485",
                uri="/digest/",
                algorithm=MD5,
                response="814bc0d6644fa1202650e2c404460a21",
                qop=auth,
                nc=00000001,
                cnonce="3da69c14300e446b"
```

La información que se manda es *responde* que en este caso esta cifrada usando md5 y que se calcula de la siguiente manera:

- Se calcula el md5 del nombre de usuario, del dominio (realm) y la contraseña, la llamamos HA1.
- Se calcula el md5 del método de la petición (por ejemplo GET) y de la uri a la que estamos accediendo, la llamamos HA2.
- El reultado que se manda es el md5 de HA1, un número aleatorio (nonce), el contador de peticiones (nc), el qop y el HA2.

Una vez que lo recibe el servidor, puede hacer la misma operación y comprobar si la información que se ha enviado es válida, con lo que se permitiría el acceso.

Ejercicio:

1. Crea dos subdirectorios en el host virtual defaul que se llamen `grupo1` y `grupo2`. Crea varios usuarios con la utilidad `htdigest`, asignando a cada uno un dominio distinto (`domgrupo1` y `domgrupo2`). Configura los directorios para que al primero grupo1 sólo puedan acceder los usuarios del dominio domgrupo1, y el directorio grupo2 solo accedan los usuarios del dominio domgrupo2.

3.7 Ejercicio: Configuración de apache mediante archivo .htaccess

Un fichero `.htaccess` (hypertext access), también conocido como archivo de configuración distribuida, es un fichero especial, popularizado por el Servidor HTTP Apache que nos permite definir diferentes directivas de configuración para cada directorio (con sus respectivos subdirectorios) sin necesidad de editar el archivo de configuración principal de Apache.

Para permitir el uso de los ficheros .htaccess o restringir las directivas que se puedn aplicar usamos ela directiva AllowOverride, que puede ir acompañada de una o varias opciones: `All`, `AuthConfig`, `FileInfo`, `Indexes`, `Limit`, ... Estudia para que sirve cada una de las opciones.

Ejercicios

Utiliza una cuenta de un servidor remoto para comprobar el uso de `.htacces`. Crea un directorio dentro de html_public y crea un fichero .htaccess que nos permita:

1. Deshabilitar la opción de listar los ficheros en ese directorio.
2. Hacer que la página entrada.html se visualice por defecto.
3. Hacer que los ficheros txt no sean accesibles.
4. (Se debe hacer en local) Redireccionar a una página (por ejemplo la web del instituto) excepto a unas determinadas IP
5. Crear una lista de IPs prohibidas
6. Protege tu directorio y ficheros con autentificación básica
7. Crear una página personalizada para cada tipo de error
8. Crea una redirección permanente: cuando entremos en este directorio salte a www,google.es
9. Permitir la entrada desde un cliente en concreto (utilizando el nombre del host), si no se entra desde esa máquina, pedir autentificación.
10. Usar negociación de contenidos: tener dos páginas en distinto idioma y configurar en el .htaccess que idioma es el prioritario.

3.8 Ejercicio: Módulos en apache

3.8.1 Directorios web para cada usuario (public_html)

El módulo userdir permite que cada usuario del sistema tenga la posibilidad de tener un directorio (por defecto se llama `public_html`) donde alojar su página web.

Ejercicios

1. Activa el módulo y comprueba su funcionamiento.
2. Comprueba las opciones configuradas para los directorios public_html.
3. Cambia el nombre de directorio public_html por otro nombre.
4. Publica una página de un usuario, y accede a la misma.

3.8.2 Creación de un servidor WebDAV

WebDAV («*Edición y versionado distribuidos sobre la web*») es un protocolo para hacer que la www sea un medio legible y editable. Este protocolo proporciona funcionalidades para crear, cambiar y mover documentos en un servidor remoto (típicamente un servidor web). Esto se utiliza sobre todo para permitir la edición de los documentos que sirve un servidor web, pero puede también aplicarse a sistemas de almacenamiento generales basados en web, que pueden ser accedidos desde cualquier lugar. La mayoría de los sistemas operativos modernos proporcionan soporte para WebDAV, haciendo que los ficheros de un servidor WebDAV aparezcan como almacenados en un directorio local.

Configuración de un servidor WebDAV

Para crear un directorio en nuestro servidor Web que pueda ser accesible por medio de un cliente WebDAV debemos activar los módulos **dav** y **dav_fs**.

Lo primero es indicar el nombre de la base de datos de lock que se utilizará, mediante la directiva DAVLockDB. Es importante tener especial cuidado con esta directiva, ya que es frecuente fuente de errores:

```
DavLockDB /tmp/DAVLockDB
```

Lo que indica la directiva no es ni el nombre de un archivo ni el de una carpeta, si no la parte inicial del nombre de un archivo. El módulo creará un archivo de nombre `DAVLockDB.orig` y otro de nombre `DAVLockDB.xxxxx` dentro de la carpeta indicada, para lo cual es necesario que el usuario *«Apache»* tenga permisos de escritura en ella.

A continuación creamos una sección directory para el directorio que queremos acceder por WebDav y activar el modo WebDav con la directiva `dav on`. Además por seguridad se debe autentificar el acceso, por lo que quedaría parecido a esto:

```
DavLockDB /tmp/DavLock
```

](Directory /var/www/webdav> dav on Options Indexes FollowSymLinks MultiViews AllowOverride None Require all granted AuthType digest AuthUserFile «/etc/apache2/digest.txt» AuthName «Dominio» Require valid-user](/Directory>

Por último prueba un cliente WebDAV en Linux y otro en Windows y comprueba el funcionamiento.

3.8.3 Módulo rewrite

El módulo **rewrite** nos va a permitir acceder a una URL e internamente estar accediendo a otra. Ayudado por los ficheros `.htaccess`, el módulo rewrite nos va a ayudar a formar URL amigables que son más consideradas por los motores de búsquedas y mejor recordadas por los humanos. Por ejemplo estas URL:

```
www.dominio.com/articulos/muestra.php?id=23
www.dominio.com/pueblos/pueblo.php?nombre=torrelodones
```

Es mucho mejor escribirlas como:

```
www.dominio.com/articulos/23.php
www.dominio.com/pueblos/torrelodones.php
```

Ejemplo 1: Reescribir URL

Si tenemos el siguiente fichero php (descargar) llamado operacion.php, podríamos usarlo de la siguiente manera:

```
http://localhost/operacion.php?op=suma&op1=6&op2=8
```

Y si queremos reescribir la URL y que usemos en vez de php html, de esta forma:

```
http://localhost/operacion.html?op=suma&op1=6&op2=8
```

Para ello activamos el `mod_rewite`, y escribimos un `.htaccess` de la siguiente manera:

```
Options FollowSymLinks
RewriteEngine On
RewriteRule ^operacion.html$ operacion.php
```

Ejemplo 2: Cambiar la extensión de los ficheros

Si queremos usar la extensión do en vez de html podríamos usar este `.htaccess`:

```
Options FollowSymLinks
RewriteEngine On
RewriteRule ^(.+).do$ $1.html [nc]
```

Esto puede ser penalizado por los motores de búsqueda ya que podemos acceder a la misma página con dos URL distintas, para solucionar esto podemos hacer una redirección:

```
RewriteRule ^(.+).do$ $1.html [r,nc]
```

Ejemplo 3: Crear URL amigables

Como habíamos visto anteriormente el fichero operacion.php se podía ejecutar de la siguiente manera:

```
http://localhost/operacion.php?op=suma&op1=6&op2=8
```

Creando una URL amigable podríamos llamar a este fichero de la siguiente manera:

```
http://localhost/suma/8/6
```

¿Cómo podemos conseguir esto?

Crea un `.htaccess` con el siguiente contenido:

```
Options FollowSymLinks
RewriteEngine On
RewriteBase /
RewriteRule ^([a-z]+)/([0-9]+)/([0-9]+)$ operacion.php?op=$1&op1=$2&op2=$3
```

Ejemplo 4: Acortar URL

Supongamos que dentro de nuestro *DocumentRoot* tenemos una carpeta búsqueda con un fichero `buscar.php` (descargar). Este fichero me permite obtener la página de búsqueda de google con el parámetro dado, de esta forma:

```
http://localhost/busqueda/buscar.php?id=hola
```

Nos gustaría poder crear una URL más corta que haga lo mismo, escribiríamos en nuestro `.htaccess` un *RewriteRule* de la siguiente forma:

```
RewriteRule ^buscar busqueda/buscar.php
```

De esta forma accederíamos por medio de la URL:

```
http://localhost/buscar?id=hola
```

Ejercicio:

Siguiendo las técnicas anteriormente vistas, realiza una reescritura de URL para que pudiéramos realizar búsquedas con URL de la siguiente manera:

```
http://localhost/buscar/hola.html
```

Ejemplo 5: Uso del RewriteCond

La directiva `RewriteCond` nos permite especificar una condición que si se cumple se ejecuta la directiva `RewriteRule` posterior. Se pueden poner varias condiciones con `RewriteCond`, en este caso cuando se cumplen todas se ejecuta la directiva `RewriteRule` posterior.

Como vemos en la documentación podemos preguntar por varios parámetros , entre los que destacamos los siguientes:

- **%{HTTP_USER_AGENT}**: Información del cliente que accede. Por ejemplo, podemos mostrar una página distinta para cada navegador:

```
RewriteCond %{HTTP_USER_AGENT} ^Mozilla
RewriteRule ^/$ /index.max.html [L]
RewriteCond %{HTTP_USER_AGENT} ^Lynx
RewriteRule ^/$ /index.min.html [L]
RewriteRule ^/$ /index.html [L]
```

- **%{QUERY_STRING}**: Guarda la cadena de parámetros de una URL dinámica.Por ejemplo:

Teníamos un fichero index.php que recibía un parámetro lang, para traducir el mensaje de bienvenida:

```
http://localhost/index.php?lang=es
```

Actualmente hemos cambiado la forma de traducir, y se han creado distintos directorios para cada idioma y dentro un index.php con el mensaje traducido:

```
http://localhost/es/index.php
```

Sin embargo se quiere seguir utilizando la misma forma de traducir:

```
RewriteCond %{QUERY_STRING} lang=(.*)
RewriteRule ^index.php$ /%1/$1
```

- **%{REMOTE_ADDR}**: Dirección de destino. Por ejemplo puedo denegar el acceso a una dirección:

```
RewriteCond %{REMOTE_ADDR} 145.164.1.8
RewriteRule ^(.*)$ / [R,NC,L]
```

También podemos controlar la reescritura de URL según la hora y la fecha, para saber más lee este artículo.

- **%{HTTP_REFERER}**: Guarda la URL que accede a nuestra página y %{REQUEST_URI} guarda la URI, URL sin nombre de dominio. Podemos evitar el Hot_Linking, o uso de recursos de tu servidor desde otra web. Por ejemplo, un caso muy común es usar imágenes alojadas en tu servidor puestas en otras web. Para ello podemos escribir el siguiente `.htaccess`:

```
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !^http://(www\.)?dominio\.com/ [NC]
RewriteCond %{REQUEST_URI} !hotlink\.(gif|png) [NC]
RewriteRule .*\.(gif|jpg|png)$ http://www.dominio.com/image/hotlink.png [NC]
```

En el anterior ejemplo el primer `RewriteCond` permite la solicitud directa pero no desde otras páginas (referrer vacío). La siguiente línea indica que si el navegador ha enviado una cabecera `Referrer` y esta no contiene la palabra «dominio.com» se ejecutará el `RewriteRule`. La ultima instrucción `RewriteCond` indica que si en la url solicitada se encuentra el nombre de la imagen «hotlink» no se realizará el `RewriteRule`; esto se pone porque la imagen hotlink.png va a ser la que vamos a usar en `RewriteRule` y si no ponemos este `RewriteCond` también sería redirigida la solicitud a esta imagen. La última instrucción del ejemplo es el `RewriteRule` que indica que cualquier solicitud a una imagen desde otro referrer será reescrita en el servidor hacia la imagen hotlink.png y esta será la imagen que se vea en la web que te esté intentando robar la imagen.

Ejercicio:

Realiza un `.htaccess` para evitar el hot-linking. Puedes usar esta esta imagen para realizar el ejercicio.

Ejemplo 6: URL amigables con WordPress

Ejercicio: Instala wordpress en tu servidor con el módulo rewrite desactivado, comprueba que las URL no son amigables. Activa el módulo y a continuación configura el blog para que tenga URL amigables (Settings->Permalink).

CAPÍTULO 4

Servidor DNS

En este bloque vamos a estudiar cómo funciona el protocolo DNS, y vamos a configurar distisntos servidores DNS. El DNS se utiliza para distintos propósitos. Los más comunes son:

- Resolución de nombres: Dado el nombre completo de un host obtener su dirección IP.
- Resolución inversa de direcciones: Es el mecanismo inverso al anterior. Consiste en, dada una dirección IP, obtener el nombre asociado a la misma.
- Resolución de servidores de correo: Dado un nombre de dominio, obtener el servidor a través del cual debe realizarse la entrega del correo electrónico.

Los contenidos que vamos a estudiar en este bloque serán:

- Cómo funciona el protocolo DNS
- Realizar consultas a los servidores DNS
- Estudio de distintos servidores DNS: DNS Windows Server, dnsmasq, bind9
- Servidores DNS esclavos
- Configuración de subdominios, delegación de subdominios.
- Servidores DNS dinámicos

Indice

4.1 Enlaces interesantes

Protocolo DNS

- ¿Cómo funciona el DNS?
- Ficheros importantes en la resolución de nombres

DNSmasq

- Fundamentos del servicio DNS

- HowTodnsmasq

bind9

- Servidor dns: Bind9
- Esquema: Servidor DNS [jpg]
- Servidores DNS esclavos (1ª parte)
- Servidores DNS esclavos (2ª parte)
- Configurar subdominios en bind9
- Servidor DNS dinámico
- Esquema: Proceso DNS dinámico [jpg]
- DNS dinámico (Desde lo alto del cerro)
- Zonas inversas IPv6

4.2 Ejercicio: Consultas DNS

dig

dig es una herramienta que permite hacer consultas a un servidor DNS desde la línea de comandos, es el sustituto de los programas nslookup y host. La sintaxis es:

```
dig [-t tipo de registro] [@servidor DNS] Consulta DNS
```

El tipo de registro por defecto es ADDRESS y el servidor DNS por defecto el definido en `/etc/resolv.conf`.

Nota: si no funciona el comndo dig, instala el paquete dnsutils que lo incluye.

nslookup

Como realizar consultas DNS con el nslookup de Windows.

Utilizando el comando dig/nslookup realiza las siguientes consultas al servidor DNS:

1. Preguntas a registros del tipo A: Obtén la dirección ip de los siguientes dominios:

   www.gonzalonazareno.org www.eltiempo.es www.us.es es.wikipedia.org www.ubuntu.com

2. Preguntas a registros tipo NS: Obtén la dirección y los servidor DNS que corresponden a los siguientes dominios:

   dominio raíz com org es us.es wikipedia.org ubuntu.com

3. Preguntas a registros MX: Obtén el nombre y la dirección del ordenador al que se mandan los correos que se envían a los siguientes dominios:

   gonzalonazareno.org us.es wikipedia.org ubuntu.com

4. ¿Qué tipo de registro es el que resuelve las siguientes direcciones:

   www.josedomingo.org informatica.gonzalonazareno.org

Indica el nombre canónico de las máquinas a las que corresponden.

1. Comprueba la dirección ip y el servidor DNS asociado a `dit.gonzalonazareno.org`, desde dentro de la intranet del ciclo formativo y desde fuera. ¿Cuáles son las diferencias? ¿A qué crees que es debido?
2. Pregunta sobre resolución inversa: En clase, ¿a qué nombre corresponde la dirección ip `172.22.0.1`?

4.3 Ejercicio: DNSmasq como DNS cache/forward en una red local

El paquete `dnsmasq` permite poner en marcha un servidor DNS de una forma muy sencilla. Simplemente instalando y arrancando el servicio dnsmasq, sin realizar ningún tipo de configuración adicional, nuestro PC se convertirá en un servidor caché DNS y además, resolverá los nombres que tengamos configurados en el archivo /etc/hosts de nuestro servidor. La resolución funcionará tanto en sentido directo como en sentido inverso, es decir, resolverá la IP dado un nombre de PC y el nombre del PC dada la IP.

Queremos instalar un servidor DNS local en nuestra intranet que nos permita gestionar los nombres de las máquinas y recursos de nuestra red, vamos a instalar el servidor DNS en el mismo ordenador que tenemos instalado el servidor web. Las características del servidor DNS que queremos instalar son las siguientes:

- El servidor web (IP que tenga el servidor linux) se llama `nombredelservidor.iesgn.com`
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.iesgn.com` y que está en 192.168.1.201 (esto es ficticio)
- Además queremos nombrar a otros clientes.
- Las páginas webs que hiciste en la práctica de apache (`www.iesgn.org`,...) tienen que ser accesibles desde los clientes.

Una vez instalado, el paquete, editamos el fichero `/etc/dnsmasq.conf` y modificamos las siguientes líneas:

- Descomentamos `strict-order` para que se realicen las peticiones DNS a los servidores que aparecen en el fichero `/etc/resolv.conf` en el orden en el aparecen.
- Incluimos las interfaces de red que deben aceptar peticiones DNS, descomentando la línea interface por ejemplo: `interface=eth0`

Finalmente reiniciamos el servicio.

Advertencia:

1. Configura los clientes para que utilicen el servidor DNS que has instalado.
2. Realiza las consultas dig/nslookup desde los clientes preguntando por los siguientes:
   - Dirección de `nombredelservidor.iesgn.org`, `www.iesgn.org`, `ftp.iesgn.org`
   - La dirección IP de `www.josedomingo.org`
3. Comprueba que se puede entrar en las páginas webs

4.4 Ejercicio: Instalación y configuración del servidor bind9 en nuestra red local

Nota:

1. Desinstala el servidor dnsmasq que has instalado en la práctica anterior para que no tengas conflictos.
2. Para hacer este ejercicio vamos a suponer que nuestro ordenadores están en la red 10.0.0.0/24, siendo nuestro servidor el 10.0.0.3, y los clientes 10.0.0.4 y 10.0.0.5. Adapta este direccionamiento a tu escenario.

Queremos instalar un servidor DNS local en nuestra intranet que nos permita gestionar los nombres de las máquinas y recursos de nuestra red, vamos a instalar el servidor DNS en nuestro servidor debian. Las características del servidor DNS que queremos instalar son las siguientes:

1. Vamos a crear una zona para el dominio: `iesgn.org`
2. Vamos a crear una zona de resolución inversa.
3. Vamos a tener los siguientes FQDN

- El servidor DNS se llama `nombredelservidor.iesgn.org`
- Vamos a suponer que tenemos un servidor para recibir los correos que se llame `correo.iesgn.org` y que está en 10.0.0.200 (esto es ficticio)
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.iesgn.org` y que está en 10.0.0.201 (esto es ficticio)
- Además queremos nombrar a varios clientes.
- Suponemos que tenemos un servidor web con las páginas: `www.iesgn.org` y `departementos.iesgn.org`

Advertencia:

1. Configura el servidor DNS con los registros A, CNAME, MX y NS necesarios, configura el SOA.
2. Configura los clientes para que su DNS sea el servidor Debian, debes indicar en la configuración de red del cliente como DNS primario la ip del servidor linux.
3. Realiza las consultas dig/neslookup desde los clientes preguntando por los siguientes:

- Dirección de `nombredelservidor.iesgn.org`,‘‘www.iesgn.org‘‘,‘‘ftp.iesgn.org‘‘
- El servidor DNS que tiene configurado la zona del dominio `iesgn.org`
- El servidor de correo configurado para `iesgn.org`
- La dirección IP de `www.josedomingo.org`
- Un resolución inversa

4.5 Ejercicio: Instalación y configuración de un servidor DNS esclavo

Nota: Suponemos que tenemos instalado el servidor DNS del ejercicio anterior.

Actualmente tenemos instalado un servidor DNS que tiene autoridad sobre la zona `iesgn.org` y sobre la zona de resolución inversa correspondiente. Este servidor va a funcionar como **DNS maestro**. Vamos a instalar un nuevo servidor DNS que va a estar configurado como **DNS esclavo** del anterior, donde se van a ir copiando periódicamente las zonas del DNS maestro. Suponemos que el nombre del servidor DNS esclavo se va llamar `nombredelesclavo.iesgn.org`.

Modifica la configuración del servidor DNS actual para que funcione como maestro de las zonas que tiene definida. Instala y configura un nuevo servidor DNS para que funcione como DNS esclavo del anterior. Y realiza las siguientes comprobaciones:

Advertencia:

1. Entrega la configuración de las zonas del maestro y del esclavo.
2. Comprueba si las zonas definidas en el maestro tienen algún error con el comando adecuado.
3. Comprueba si la configuración de named.conf tiene algún error con el comando adecuado.
4. Reinicia los servidores y comprueba en los logs si hay algún error. **No olvides incrementar el número de serie en el registro SOA si has modificado la zona en el maestro**.
5. Muestra la salida del log donde se demuestra que se ha realizado la transferencia de zona.
6. Realiza una consulta con dig tanto al maestro como al esclavo para comprobar que las respuestas son autorizadas. ¿En qué te tienes que fijar?
7. Configura un cliente para que utilice los dos servidores como servidores DNS.
8. Solicita una copia completa de la zona desde el cliente ¿qué tiene que ocurrir?. Solicita una copia completa desde el esclavo ¿qué tiene que ocurrir?
9. Realiza una consulta desde el cliente y comprueba que servidor está respondiendo.
10. Posteriormente apaga el servidor maestro y vuelve a realizar una consulta desde el cliente ¿quién responde?

4.6 Ejercicio: Configuración de subdominios virtuales con bind9

Nota: Suponemos que tenemos instalado el servidor DNS del ejercicio anterior.

Tenemos un servidor DNS que gestiona la zona correspondiente al nombre de dominio `iesgn.org`, queremos configurar dicho servidor para crear el subdominio `informatica.iesgn.org`. Los nombres que vamos a tener en ese subdominio son los siguientes:

- `www.informatica.iesgn.org` corresponde a un sitio web que está en la dirección 10.0.0.100.
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.informatica.iesgn.org` y que está en la misma máquina.

Advertencia:

1. Configura el servidor DNS para poder tener el subdominio virtual `informatica.iesgn.org`.
2. Realiza las consultas dig/neslookup desde los clientes preguntando por los siguientes:
   - Dirección de `www.informatica.iesgn.org`, `ftp.informatica.iesgn.org`
   - El servidor DNS que tiene configurado la zona del dominio `informatica.iesgn.org`. Comprueba que es el mismo que el de la zona `iesgn.org`.

4.7 Ejercicio: Delegación de subdominios con bind9

Nota: Suponemos que tenemos instalado el servidor DNS del ejercicio anterior.

Tenemos un servidor DNS que gestiona la zona correspondiente al nombre de dominio `iesgn.org`, en esta ocasión queremos delegar el subdominio `informatica.iesgn.org` para que lo gestione otro servidor DNS. Por lo tanto tenemos un escenario con dos servidores DNS:

- `nombredelservidor.iesgn.org`, es servidor DNS autorizado para la zona `iesgn.org` y tiene asignado la dirección 10.0.0.3.
- `servidor_subdomio.informatica.iesgn.org`, es el servidor DNS para la zona `informatica.iesgn.org` y, suponemos que tiene asignado la dirección 10.0.0.50.

Los nombres que vamos a tener en ese subdominio son los siguientes:

- `www.informatica.iesgn.org` corresponde a un sitio web que está en la dirección 10.0.0.100.
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.informatica.iesgn.org` y que está en la misma máquina.
- Vamos a suponer que tenemos un servidor para recibir los correos que se llame `correo.informatica.iesgn.org` y que está en 10.0.0.51.

Advertencia:

1. Configura el primer servidor DNS para poder tener el subdominio virtual `informatica.iesgn.org`.
2. Configura el segundo servidor DNS con los registros A, CNAME, MX y NS necesarios para el subdominio `informatica.iesgn.org`.
3. Realiza las consultas dig/neslookup desde los clientes preguntando por los siguientes:
   - Dirección de `www.informatica.iesgn.org`, `ftp.informatica.iesgn.org`
   - El servidor DNS que tiene configurado la zona del dominio `informatica.iesgn.org`. ¿Es el mismo que el servidor DNS con autoridad para la zona `iesgn.org`?
   - El servidor de correo configurado para `informatica.iesgn.org`

4.8 Ejercicio: Instalación y configuración de un servidor DNS dinámico

Nota: Suponemos que tenemos instalado el servidor DNS del ejercicio anterior.

Suponemos que actualmente tenemos instalado un servidor DNS caché que da servicio a los ordenadores de nuestra intranet y además actúa como servidor maestro (master) de un dominio DNS (`iesgn.org`), de forma que todos los equipos de la red local tengan un nombre DNS completo o FQHN (Full Qualified Host Name). Por otra parte, tenemos instalado en la misma máquina un servidor DHCP para que asigne direcciones IPv4 únicas a los equipos de la red local y les facilite el resto de parámetros necesarios para tengan conectividad y salida a Internet.

Las características del servidor DNS serán:

- El nombre del servidor DNS para la zona `iesgn.org` sera `nombredelservidor.iesgn.org`.
- En un primer momento el único nombre que resuelve nuestro servidor será el suyo propio, con la dirección 192.168.2.1 *(Cambia el direccionamiento según tu escenario).*

Las características del servidor DHCP instalado serán:

- Tiempo de concesión: 1 mes

- Rango de direcciones: 192.168.2.100 - 192.168.2.150
- Puerta de enlace: 192.168.2.1
- Servidores DNS: 192.168.2.1

Advertencia:

1. Entrega la configuración en el servidor DNS necesaria para que tenga la funcionalidad de de DNS dinámico.
2. Muestra la configuración necesaria en el servidor DHCP para comunicarse con el DNS.
3. Configura un cliente de forma dinámica para que tome una configuración ofrecida por el servidor DHCP y comprueba que su nombre se ha incluido en la zona correspondiente en el servidor DNS: visualiza los logs que no informan de ello y realiza una consulta al servidor DNS preguntando por su nombre.
4. Fuerza a que un cliente cambie de DNS y comprueba que la modificación se ha comunicado al DNS.

CAPÍTULO 5

Servidor FTP

FTP (siglas en inglés de File Transfer Protocol, "Protocolo de Transferencia de Archivos") en informática, es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP (Transmission Control Protocol), basado en la arquitectura cliente-servidor. Desde un equipo cliente se puede conectar a un servidor para descargar archivos desde él o para enviarle archivos, independientemente del sistema operativo utilizado en cada equipo.

Los contenidos que vamos a estudiar del servidor FTP, nos van a pobilitar responder las siguientes preguntas:

1. ¿Qué signifcan las siglas FTP? ¿Para qué sirve dicho protocolo? ¿Qué incoveniente tiene su uso?
2. ¿Qué puertos utiliza el prótocolo? ¿Para qué sirve cada uno?
3. ¿Qué tipos de acceso podemos configurar en un servidor FTP?
4. ¿Qué modos de conexión existen en el prótocolo FTP? ¿En qué consiste cada uno y cuál es la diferencia?
5. Índica los tipos de transferncia que podemos hacer. ¿Qué tipo utilizarías si vas a transferir un fichero `odt`?
6. ¿Qué tipo de clientes FTP existen?
7. Si usas un cliente FTP de línea de comandos, que comando tendrías que ejecutar para conectarte al servidor `ftp.gonzalonazareno.org` y subir un fichero `practica.pdf`.

Indice

5.1 Enlaces interesantes

- Configuración básica de proFTPd en debian
- El protocolo FTP
- ProFTPd
- ProFTPd: Manual de instalación y configuración
- net2ftp: Cliente FTP basado en la web

- Cómo instalar Proftpd con soporte MySQL
- Servidor ftp Windows 2008:Vídeo 1 Vídeo 2

5.2 Ejercicio: Instalación de proFTPd y uso de clientes FTP

Instala el servidor proFTPd y comprueba su funcionamiento desde un cliente FTP gráfico: filezilla, y un cliente FTP de texto: ftp.

Guía de comandos FTP::

```
    Comando y argumentos    Acción que realiza
    open servidor       Inicia una conexión con un servidor FTP.
    close o disconnect  Finaliza una conexión FTP sin cerrar el programa cliente.
    bye o quit      Finaliza una conexión FTP y la sesión de trabajo con el programa␣
↪cliente.
    cd directorio       Cambia el directorio de trabajo en el servidor.
    delete archivo      Borra un archivo en el servidor
    mdelete patrón      Borra múltiples archivos basado en un patrón que se aplica al␣
↪nombre.
    dir             Muestra el contenido del directorio en el que estamos en el␣
↪servidor.
    get archivo         Obtiene un archivo
    noop No Operation   Se le comunica al servidor que el cliente está en modo de no␣
↪operación, el servidor usualmente responde con un «ZZZ» y refresca el contador de␣
↪tiempo inactivo del usuario.
    mget archivos       Obtiene múltiples archivos
    hash            Activa la impresión de caracteres # a medida que se transfieren␣
↪archivos, a modo de barra de progreso.
    lcd directorio      Cambia el directorio de trabajo local.
    ls          Muestra el contenido del directorio en el servidor.
    prompt          Activa/desactiva la confirmación por parte del usuario de la␣
↪ejecución de comandos. Por ejemplo al borrar múltiples archivos.
    put archivo         Envía un archivo al directorio activo del servidor.
    mput archivos       Envía múltiples archivos.
    pwd             Muestra el directorio activo en el servidor.
    rename archivo      Cambia el nombre a un archivo en el servidor.
    rmdir directorio    Elimina un directorio en el servidor si ese directorio está␣
↪vacío.
    status          Muestra el estado actual de la conexión.
    bin o binary        Activa el modo de transferencia binario.
    ascii           Activa el modo de transferencia en modo texto ASCII.
    !           Permite salir a línea de comandos temporalmente sin cortar la␣
↪conexión. Para volver, teclear exit en la línea de comandos.
    ? nombre de comando     Muestra la información relativa al comando.
    ? o help        Muestra una lista de los comandos disponibles.
    append archivo      Continua una descarga que se ha cortado previamente.
    bell            Activa/desactiva la reproducción de un sonido cuando ha terminado␣
↪cualquier proceso de transferencia de archivos.
    glob            Activa/desactiva la visualización de nombres largos de nuestro PC.
    literal         Con esta orden se pueden ejecutar comandos del servidor de forma␣
↪remota. Para saber los disponibles se utiliza: literal help.
    mkdir           Crea el directorio indicado de forma remota.
    quote           Hace la misma función que literal.
    send archivo        Envía el archivo indicado al directorio activo del servidor.
    user            Para cambiar nuestro nombre de usuario y contraseña sin necesidad␣
↪de salir de la sesión ftp.
```

Modifica la configuración del servidor para que los usuarios sólo puedan entrar en su directorio «Documentos».

Nota: Todos los accesos al servidor FTP lo vamos a hacer utilizando su nombre, por ejemplo ftp.iesgn.org, por lo tanto debes configurar el servidor BIND9 en el servidor para que todos los clientes conozcan este nombre.

Siguiendo las indicación de la documentación suministrada, configura el servidor proFTPd para crear un servidor FTP anónimo de sólo lectura.

Cuando termines, aunque no sea recomendable, configura el servidor proftpd para hacerlo anónimo y de lectura y escritura.

CAPÍTULO 6

Gestión de peticiones y rendimiento en servidores Web

Los servidores web pueden ser configurado para manejar las peticiones de diferente forma, desde el punto de vista en que son creados y manejados los subprocesos necesarios que atienden a cada cliente conectado a este. En esta unidad vamos a explicar los MPM (Módulos de multiprocesamiento) que nos permiten configurar el servidor Web para gestionar las peticiones que llegan al servidor.

Vamos a estudiar tambíen las distintas configuraciones que podemos realizar para que los servidores web sean capaces de servir páginas realizadas en PHP y Python y vamos a estudiar las diferencias de rendimientos (respuestas por segundos) que podemos obtener utilizando las distintas configuraciones. Por último estudiaremos diferentes aplicaciones que pueden mejorar el rendimiento de nuetro servidor: aceleradores PHP, memcache, varnish, ...

Para terminar la unidad vamos a hacer un estudio comparativo sobre el rendimiento entre distintos servidores web.

6.1 Enlaces interesantes

Módulos de multiprocesamiento en Apache 2.4

- Módulos de multiprocesamiento (MPMs)
- Entendiendo los modos multiproceso de Apache
- Diferencias entre apache prefork, event y worker
- Apache2 us ngninx
- How to Configure nginx for Optimized Performance

6.2 Ejercicio: Gestión de peticiones

6.2.1 Módulos de Multiprocesamiento (MPMs) en Apache 2.4

Por defecto apache2 se configura con el MPM event, podemos ver el MPM que estamos utilizando con la siguiente instrucción::

```
# apachectl -V
...
Server MPM:     event
...
```

Para cambiar de MPM tenemos que desactivar el actual y activar el nuevo módulo::

```
# a2dismod mpm_event
# a2enmod mpm_prefork
# service apache2 restart

# apachectl -V
...
Server MPM:     prefork
...
```

Las directivas de configuración de los distintos MPM

En `/etc/apache2/mods-availables/mpm_prefork.conf`::

Directivas de control de prefork:

```
StartServers              5
MinSpareServers           5
MaxSpareServers          10
MaxRequestWorkers       150
MaxConnectionsPerChild    0
```

En `/etc/apache2/mods-availables/mpm_worker.conf`:

Directivas de control de worker:

```
StartServers            2
MinSpareThreads         25
MaxSpareThreads         75
ThreadLimit             64
ThreadsPerChild         25
MaxRequestWorkers       150
MaxConnectionsPerChild  0
```

En `/etc/apache2/mods-availables/mpm_event.conf`:

Directivas de control de event:

```
StartServers              2
MinSpareThreads          25
MaxSpareThreads          75
ThreadLimit              64
ThreadsPerChild          25
MaxRequestWorkers       150
MaxConnectionsPerChild    0
```

6.2.2 Gestión de peticiones en nginx

Procesos worker

La directiva `worker_processes` nos indica el número de procesos que van a responder peticiones. El valor de `worker_processes`, se suele definir al mismo número de CPUs que tenga el equipo, o como mucho, el doble.

Esto se hace así porque, al ser un servidor web asíncrono, cada proceso se puede encargar de muchas peticiones y por lo tanto no tiene sentido tener más procesos que CPUs capaces de ejecutar código.

Para indicar un worker por cada core de la CPU:

```
worker_processes auto;
```

- Comprueba la configuración por defecto de nginx y comprueba cuantos procesos worker se están ejecutando.

Conexiones por proceso

La opción `worker_connections` establece el número máximo de conexiones que cada proceso worker puede procesar a la vez. Es recomendable aumentar este valor si nuestra web tiene un elevado tráfico. El valor por defecto es de 768.

El número máximo de clientes que Nginx puede manejar viene determinado por multiplicar el valor indicado en `worker_processes` por el indicado en `worker_connections`.

Además podemos usar `multi_accept` con el fin de que un worker acepte todas las nuevas conexiones al mismo tiempo.

- Comprueba la configuración por defecto de nginx.

6.3 Ejecución de script PHP

6.3.1 Apache2 y módulo PHP

Instalamos apache2 y el módulo que permite que los procesos de apache2 sean capaz de ejecutar el código PHP:

```
apt install apache2 php7.0 libapache2-mod-php7.0
```

Cuando hacemos la instalación se desactiva el MPM `event` y se activa el `prefork`:

```
...
Module mpm_event disabled.
Enabling module mpm_prefork.
apache2_switch_mpm Switch to prefork
...
```

Si queremos desactivar el módulo PHP de apache2:

```
apt remove libapache2-mod-php7.0
```

Y activamos el módulo `event`:

```
a2dismod mpm_prefork
a2enmod mpm_event
```

La configuración de php está dividida según desde se use:

- `/etc/php/7.0/cli`: Configuración de php para `php7.0-cli`, cuando se utiliza php desde la línea de comandos.
- `/etc/php/7.0/apache2`: Configuración de php para apache2 cuando utiliza el módulo.
- `/etc/php/7.0/fpm`: Configuración de php para php-fpm
- `/etc/php/7.0/mods-available`: Módulos disponibles de php que puedes estar configurados en cualquiera de los escenarios anteriores.

Si nos fijamos en la configuración de php para apache2:

- `/etc/php/7.0/apache2/conf.d`: Módulos instalados en esta configuración de php (enlaces simbólicos a `/etc/php/7.0/mods-available`).
- `/etc/php/7.0/apache2/php.ini`: Configuración de php para este escenario.

6.3.2 PHP-FPM

FPM (FastCGI Process Manager) es una implementación alternativa al PHP FastCGI. FPM se encarga de interpretar código PHP. Aunque normalmente se utiliza junto a un servidor web (Apache2 o ngnix) vamos a hacer en primer lugar una instalación del proceso y vamos a estudiar algunos parámetros de configuración y estudiar su funcionamiento.

Para instalarlo en Debian 9:

```
apt install php7.0-fpm php7.0
```

Configuración

Con esto hemos instalado php 7.0 y php-fpm. Veamos primeros algunos ficheros de configuración de php:

Si nos fijamos en la configuración de php para php-fpm:

- `/etc/php/7.0/fpm/conf.d`: Módulos instalados en esta configuración de php (enlaces simbólicos a `/etc/php/7.0/mods-available`).
- `/etc/php/7.0/fpm/php-fpm.conf`: Configuración general de php-fpm.
- `/etc/php/7.0/fpm/php.ini`: Configuración de php para este escenario.
- `/etc/php/7.0/fpm/pool.d`: Directorio con distintos pool de configuración. Cada aplicación puede tener una configuración distinta (procesos distintos) de php-fpm.

Por defecto tenemos un pool cuya configuración la encontramos en `/etc/php/7.0/fpm/pool.d/www.conf`, en este fichero podemos configurar muchos parámetros, los más importantes son:

- `[www]`: Es el nombre del pool, si tenemos varios, cada uno tiene que tener un nombre.
- `user` y `grorup`: Usuario y grupo con el que se va ejecutar los procesos.
- `listen`: Se indica el socket unix o el socket TCP donde van a escuchar los procesos:
  - Por defecto, escucha por un socket unix: `listen = /run/php/php7.0-fpm.sock`
  - Si queremos que escuche por un socket TCP: `listen = 127.0.0.1:9000`
  - En el caso en que queramos que escuche en cualquier dirección: `listen = 9000`
- Directivas de procesamiento, gestión de procesos:
  - `pm`: Por defecto igual a `dynamic` (el número de procesos se crean y destruyen de forma dinámica). Otros valores: `static` o `ondemand`.
  - Otras directivas: `pm.max_children`, `pm.start_servers`, `pm.min_spare_servers`,...
- `pm.status_path = /status`: No es necesaria, pero vamos a activar la URL de `status` para comprobar el estado del proceso.

Por último reiniciamos el servicio:

```
systemctl restart php7.0-fpm
```

Pruebas de funcionamiento

1. Suponemos que tenemos configurado por defecto, por lo tanto los procesos están escuchando en un socket UNIX:

```
listen = /run/php/php7.0-fpm.sock
```

Para enviar ficheros php a los procesos para su interpretación vamos a utilizar el programa `cgi-fcgi`:

```
apt-get install libfcgi0ldbl
```

Y a continuación accedemos a la URL `/status`, para ello:

```
SCRIPT_NAME=/status SCRIPT_FILENAME=/status REQUEST_METHOD=GET cgi-fcgi -bind -
↪connect /run/php/php7.0-fpm.sock

Expires: Thu, 01 Jan 1970 00:00:00 GMT
Cache-Control: no-cache, no-store, must-revalidate, max-age=0
Content-type: text/plain;charset=UTF-8

pool:                 www
process manager:      dynamic
start time:           13/Nov/2017:19:32:50 +0000
start since:          38
accepted conn:        6
listen queue:         0
max listen queue:     0
listen queue len:     0
idle processes:       1
active processes:     1
total processes:      2
max active processes: 1
max children reached: 0
slow requests:        0
```

Si queremos ejecutar un fichero php, vamos a crear un directorio `/var/www` y vamos a guardar un fichero `holamundo.php` con el siguiente contenido:

```
<?php echo "Hola Mundo!!!";?>
```

A continuación vamos a indicar el directorio de trabajo en el fichero `/etc/php/7.0/fpm/pool.d/www.conf`:

```
chroot = /var/www
```

Inicializamos el servicio:

```
systemctl restart php7.0-fpm
```

Y podríamos ejecutar el fichero de la siguiente manera:

```
SCRIPT_NAME=/holamundo.php SCRIPT_FILENAME=/holamundo.php REQUEST_METHOD=GET cgi-
↪fcgi -bind -connect /run/php/php7.0-fpm.sock

Content-type: text/html; charset=UTF-8

Hola Mundo!!!
```

2. Si suponemos que hemos configurado php-fpm para que escuche en un socket TCP:

```
listen = 127.0.0.1:9000
```

Para realizar las pruebas que hemos probado anteriormente:

```
SCRIPT_NAME=/status SCRIPT_FILENAME=/status REQUEST_METHOD=GET cgi-fcgi -bind -
↪connect 127.0.0.1:9000

SCRIPT_NAME=/holamundo.php SCRIPT_FILENAME=/holamundo.php REQUEST_METHOD=GET cgi-
↪fcgi -bind -connect 127.0.0.1:9000
```

Configuración de Apache2 con php-fpm

Necesito activar los siguientes módulos_

```
a2enmod proxy proxy_fcgi
```

Activarlo para cada virtualhost

Podemos hacerlo de dos maneras:

- Si php-fpm está escuchando en un socket TCP:

```
ProxyPassMatch ^/(.*\.php)$ fcgi://127.0.0.1:9000/var/www/html/$1
```

- Si php-fpm está escuchando en un socket UNIX:

```
ProxyPassMatch ^/(.*\.php)$ unix:/run/php/php7.0-fpm.sock|fcgi://127.0.0.1/var/
↪www/html
```

Otra forma de hacerlo es la siguiente:

- Si php-fpm está escuchando en un socket TCP:

```
<FilesMatch "\.php$">
    SetHandler "proxy:fcgi://127.0.0.1:9000"
</FilesMatch>
```

- Si php-fpm está escuchando en un socket UNIX:

```
<FilesMatch "\.php$">
    SetHandler "proxy:unix:/run/php/php7.0-fpm.sock|fcgi://127.0.0.1/"
</FilesMatch>
```

Activarlo para todos los virtualhost

Tenemos a nuestra disposición un fichero de configuración `php7.0-fpm` en el directorio `/etc/apache2/conf-available`. Por defecto funciona cuando php-fpm está escuchando en un socket UNIX, si escucha por un socket TCP, hay que cambiar la línea:

```
SetHandler "proxy:unix:/run/php/php7.0-fpm.sock|fcgi://localhost"
```

por esta:

```
SetHandler "proxy:fcgi://127.0.0.1:9000"
```

Por último activamos la configuración:

```
a2enconf php7.0-fpm
```

Configuración de Nginx con php-fpm

En el virtualhost descomentamos las siguientes líneas:

```
location ~ \.php$ {
            include snippets/fastcgi-php.conf;

            # With php-fpm (or other unix sockets):
            #fastcgi_pass unix:/var/run/php/php7.0-fpm.sock;
            # With php-cgi (or other tcp sockets):
            #fastcgi_pass 127.0.0.1:9000;
    }
```

Descomentando la opción si php-fpm está en un socket Unix o en un socket TCP.

6.4 Aumento de rendimiento en servidores web

Memcached

Memcached es un sistema distribuido de propósito general y que es muy usado en la actualidad por múltiples sitios web. Memcached es empleado para el almacenamiento en caché de datos u objetos en la memoria RAM, reduciendo así las necesidades de acceso a un origen de datos externo (como una base de datos o una API).

- Manual de instalación de memcached
- Como utilizar Memcached con WordPress
- Memcached para optimizar WordPress

Varnish

Varnish es un acelerador HTTP que funciona como un proxy inverso. Se sitúa por delante del servidor web, cacheando la respuesta de dicho servidor web en memoria. La próxima vez que un visitante visite la misma URL, la página será servida desde Varnish en lugar de desde el servidor web, ahorrando recursos en el backend y permitiendo más conexiones simultáneas.

- Presentación Madrid DevOps (Varnish: funcionamiento, configuración y uso)
- http://pabloroman.es/blog/2013/01/20/como-usar-varnish-para-acelerar-tu-sitio-web/
- How to: Varnish listen port 80 with systemd
- http://pabloroman.es/blog/2013/05/20/varnish-3-trucos-y-consejos/
- https://scalr-wiki.atlassian.net/wiki/display/docs/Install+Varnish+HTTP+Accelerator+with+WordPress
- http://kontsu.wordpress.com/2012/10/10/apache-2-performance-boost-with-varnish-yslow/
- Put Varnish on port 80

6.5 Ejecución de script python

6.5.1 Apache2 y módulo wsgi

Instalamos el módulo de apache2 que nos permite ejecutar código python: `libapache2-mod-wsgi`.

Veamos un ejemplo de configuración para una aplicación django. Suponemos que el fichero wsgi se encuentra en el directorio: `/var/www/html/mysite/mysite/wsgi.py` y configuramos apache2 de la siguiente manera::

```
<VirtualHost *>
    ServerName www.example.com
    DocumentRoot /var/www/html/mysite
    WSGIDaemonProcess mysite user=www-data group=www-data processes=1 threads=5␣
↪python-path=/var/www/html/mysite
    WSGIScriptAlias / /var/www/html/mysite/mysite/wsgi.py

    <Directory /var/www/html/mysite>
            WSGIProcessGroup mysite
            WSGIApplicationGroup %{GLOBAL}
            Require all granted
    </Directory>
</VirtualHost>
```

Si hemos usado un entorno virtual ecreado en el directorio `/home/debian/python`, la siguiente línea de configuración quedaría de la siguiente manera:

```
...
WSGIDaemonProcess mysite user=www-data group=www-data processes=1 threads=5 python-
↪path=/var/www/html/mysite:/home/debian/python/lib/python2.7/site-packages
...
```

6.5.2 Usando servidores wsgi

Otra forma de ejecutar código python es usar servidores de aplicación wsgi. Tenemos a nuestra disposición varios servidores: A Comparison of Web Servers for Python Based Web Applications. Si usamos el servidor web nginx está opción es la úmica disponible, pero en apache2 también la podemos usar. Realmente usamos los servidores web como proxies inversos que envían la petición python al servidor WSGI que estemos utilizando.

gunicorn

Gunicorn, también conocido como Green Unicorn (Unicornio Verde), es un servidor WSGI HTTP para Python.

Para instalarlo en Debian 9 Stretch:

```
apt install gunicorn
```

También lo podemos instalar con `pip` en un entorno virtual.

Despliegue de una aplicación django con gunicorn

Hemos creado una aplicación django en el directorio: `/home/debian/myapp` para desplegarla con gunicorn ejecutamos:

```
/home/debian/myapp# gunicorn -w 2 -b :8080 myapp.wsgi:application
[2018-01-07 18:55:34 +0000] [14329] [INFO] Starting gunicorn 19.6.0
[2018-01-07 18:55:34 +0000] [14329] [INFO] Listening at: http://0.0.0.0:8080 (14329)
[2018-01-07 18:55:34 +0000] [14329] [INFO] Using worker: sync
[2018-01-07 18:55:34 +0000] [14333] [INFO] Booting worker with pid: 14333
[2018-01-07 18:55:34 +0000] [14334] [INFO] Booting worker with pid: 14334
...
```

Con la opción `-w` indico el número de procesos que van a servir las peticiones, y con la opción `-b` indico la dirección y el puerto de escucha. Para más información: How to Deploy Python WSGI Apps Using Gunicorn.

Podemos configurar systemd para gunicorn se pueda utilizar con `systemctl`, esto lo puedes ver en el documento: Deploying Gunicorn.

Finalmente podemos configurar apache2 o nginx como proxy inversos para enviar todas las peticiones python a gunicorn (podemos hacer que el servidor web sirva el contenido estático).

uwsgi

Para instalarlo en Debian 9 Stretch:

```
apt install uwsgi
apt install uwsgi-plugin-python
```

También lo podemos instalar con `pip` en un entorno virtual.

Despliegue de una aplicación django con uwsgi

Hemos creado una aplicación django en el directorio: `/home/debian/myapp` para desplegarla con uwsgi ejecutamos:

```
uwsgi --http :8080 --plugin python --chdir /home/debian/myapp --wsgi-file myapp/wsgi.
↪py --process 4 --threads 2 --master
```

Otra alternativa es crear un fichero `.ini` de configuración, `ejemplo.ini` de la siguiente manera:

```
[uwsgi]
http = :8080
chdir = /home/debian/myapp
wsgi-file = myapp/wsgi.py
processes = 4
threads = 2
```

Y para ejecutar el servidor, simplemente:

```
uwsgi ejemplo.ini
```

De esta forma puedo tener varios ficheros de configuración del servidor uwsgi para las distintas aplicaciones python que sirva el servidor.

Podemos tener los ficheros de configuración en `/etc/uwsgi/apps-available` y para habiliatar podemos crear un enlace simbólico a estos ficheros en `/etc/uwsgi/apps-enabled`.

En el ejemplo anterior hemos usado la opción `http` para indicar que se va a devolver una respuesta HTTP, podemos usar varias opciones:

- `http`: Se comporta como un servidor http.
- `http-socket`: Si vamos a utilizar un proxy inverso usando el servidor uwsgi.

- `socket`: La respuesta ofrecida por el servidor no es HTTP, es usando el protocolo uwsgi.

Existen muchas más opciones que puedes usar: http://uwsgi-docs.readthedocs.io/en/latest/Options.html.

- Apache support
- Nginx support

6.6 El comando ab

La utilidad ab (Apache Benchmark) sirve para hacer pruebas de carga a un servidor apache. Es un programa que forma parte del apaquete `apache2-utils`.

Veamos un ejemplo::

```
ab -n 1000 -c 5 -k http://localhost/
```

El anterior comando simula 5 usuarios al mismo tiempo ahciendo 1000 peticiones al servidor web del localhost. La opción -k (keep_alive) habilita la opción de no cerrar la sesión HTTP. Veamos los resultados de este comando::

```
Benchmarking localhost (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
Completed 1000 requests
Finished 1000 requests


Server Software:        Apache/2.2.16
Server Hostname:        localhost
Server Port:            80

Document Path:          /
Document Length:        42587 bytes

Concurrency Level:      5
Time taken for tests:   0.140 seconds
Complete requests:      1000
Failed requests:        0
Write errors:           0
Keep-Alive requests:    995
Total transferred:      42903740 bytes
HTML transferred:       42587000 bytes
Requests per second:    7124.44 [#/sec] (mean)
Time per request:       0.702 [ms] (mean)
Time per request:       0.140 [ms] (mean, across all concurrent requests)
Transfer rate:          298500.90 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.1      0       3
```

(continues on next page)

(proviene de la página anterior)

```
Processing:     0    1   0.1      1       3
Waiting:        0    0   0.2      0       2
Total:          0    1   0.2      1       3

Percentage of the requests served within a certain time (ms)
  50%      1
  66%      1
  75%      1
  80%      1
  90%      1
  95%      1
  98%      1
  99%      1
 100%      3 (longest request)
```

Tenemos otra opción donde indicamos el tiempo que va a durar la prueba y el nivel de concurrencia, sería de la siguiente forma:

```
ab -t 10 -c 5 -k http://localhost/
```

En este caso se va a realizar la prueba durante 10 segundos.

6.7 Estudio de rendimiento de servidores web

Vamos a utilizar la herramienta `ab` para realizar pruebas sobre servidores web con distintas configuraciones y hacer un estudio del rendimiento de cada configuración. Para estudiar el rendimiento nos vamos a fijar en el número de peticiones respondidas por segundo por el servidor.

Hemos creado un programa benchmark.py que realiza las siguienes operaciones:

- Durante un tiempo determinado hace pruebas con la utilidad `ab`.
- Cada prueba va aumentado el nivel de concurrencia de conexiones que se hacen al servidor web.
- Cada prueba con un nivel de concurrencia determinado se hace sobre varios recursos (urls) del servidor estudiado.
- Cuando se cambia el nivel de concurrencia se reinician los servicios que se están usando.
- Finalmente por cada nivel de concurrencia te devuelve el valor medio del número respuestas por segundo que se ha conseguido en el acceso a los distintos recursos.

6.7.1 Configuración del script benchmark.py

Para cada configuración que vamos a estudiar hay que configurar el script `benchmark.py` con los valores adecuados:

- `CONN`: Lista con el nivel de concurrencia que vamos a realizar en cada prueba.
- `TITULO`: Ponemos el nombre de la configuración estudiada.
- `DURACION`: Tiempo que va a durar cada test con `ab`.
- `IP`: Dirección IP del servidor. Nosotros vamos a a hacer las pruebas desde el mismo servidor, por lo que la podemos dejar en `localhost`.
- `URLS`: Lista con las direcciones relativas que vamos a probar.
- `SERVERS`: Lista con los servicios que se van a reiniciar antes de cada prueba.

6.7.2 Ejecución del script benchmark.py

Recuerda que antes de ejecutar el script debes tener instala la aplicación `ab` que se encuentra en el paquete `apache2-utils`.

Para ejecutar el programa:

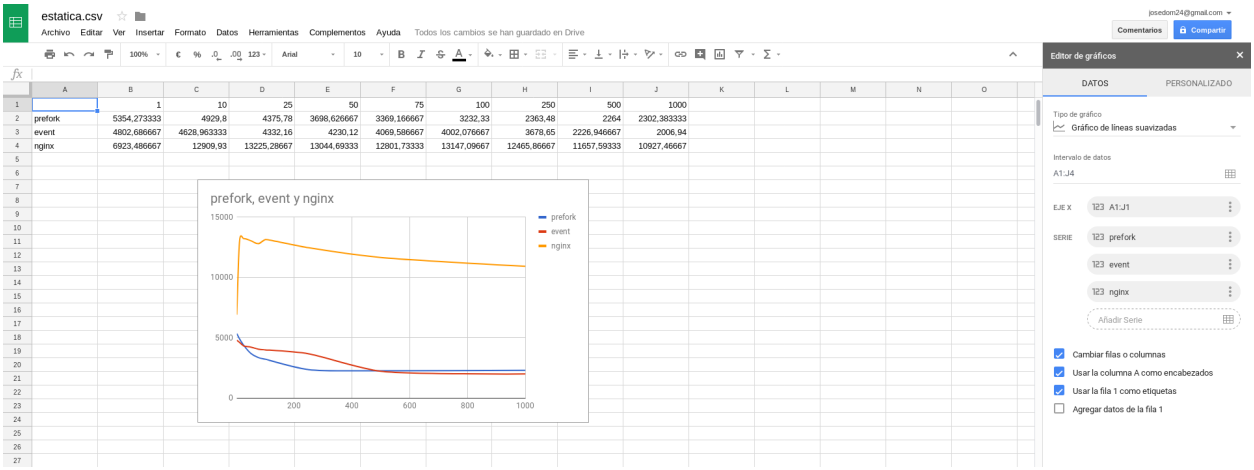
```
$ python benchmark.py

Reiniciando apache2...
Conexiones concurrentes 1
URL: http://localhost/index.html
11456.66 #/seg
URL: http://localhost/images/bg.jpg
3419.32 #/seg
URL: http://localhost/web.zip
1186.84 #/seg
Reiniciando apache2...
Conexiones concurrentes 10
URL: http://localhost/index.html
...
prefork "5354,27333333" "4929,8"    "4375,78"   "3698,62666667" "3369,16666667" "3232,
↪33"   "2363,48"   "2264,0"    "2302,38333333"
```

Realmente lo que nos interesa es la última linea donde encontramos el título, y las medias de las respuestas por segundos para cada uno de los niveles de concurrencia. Está última linea la copiamos y vamos creando un fichero csv con los datos de las distintas configuraciones estudiadas. En la primera línea ponemos los niveles de concurrencia que hemos estudiado (separados por tabuladores). Por ejemplo:

```
    1   10  25  50  75  100 250 500 1000
prefork "5354,27333333" "4929,8"    "4375,78"   "3698,62666667" "3369,16666667" "3232,
↪33"   "2363,48"   "2264,0"    "2302,38333333"
event   "4802,68666667" "4628,96333333" "4332,16"   "4230,12"   "4069,58666667" "4002,
↪07666667" "3678,65"   "2226,94666667" "2006,94"
nginx   "6923,48666667" "12909,93"  "13225,2866667" "13044,6933333" "12801,7333333"
↪"13147,0966667" "12465,8666667" "11657,5933333" "10927,4666667"
```

6.7.3 Realización de la gráfica

El fichero csv que hemos generado lo podemos subir a google drive, y desde la aplicación de hoja de cálculo podemos generar una gráfica de este tipo:



	A	B	C	D	E	F	G	H	I	J
1		1	10	25	50	75	100	250	500	1000
2	prefork	5354,273333	4929,8	4375,78	3698,626667	3369,166667	3232,33	2363,48	2264	2302,383333
3	event	4802,686667	4628,963333	4332,16	4230,12	4069,586667	4002,076667	3678,65	2226,946667	2006,94
4	nginx	6923,486667	12909,93	13225,28667	13044,69333	12801,73333	13147,09667	12465,86667	11657,59333	10927,46667

CAPÍTULO 7

Servidor de correo electrónico

Indice

7.1 Enlaces interesantes

- Cómo funciona el correo electrónico (MTA, MDA, MUA)
- Simple Mail Transfer Protocol
- Multipurpose Internet Mail Extensions
- Entender una cabecera de correo

Postfix

- Documentación: correo electrónico (Postfix)
- Documentación oficial postfix

7.2 Instalación y configuración básica de postfix

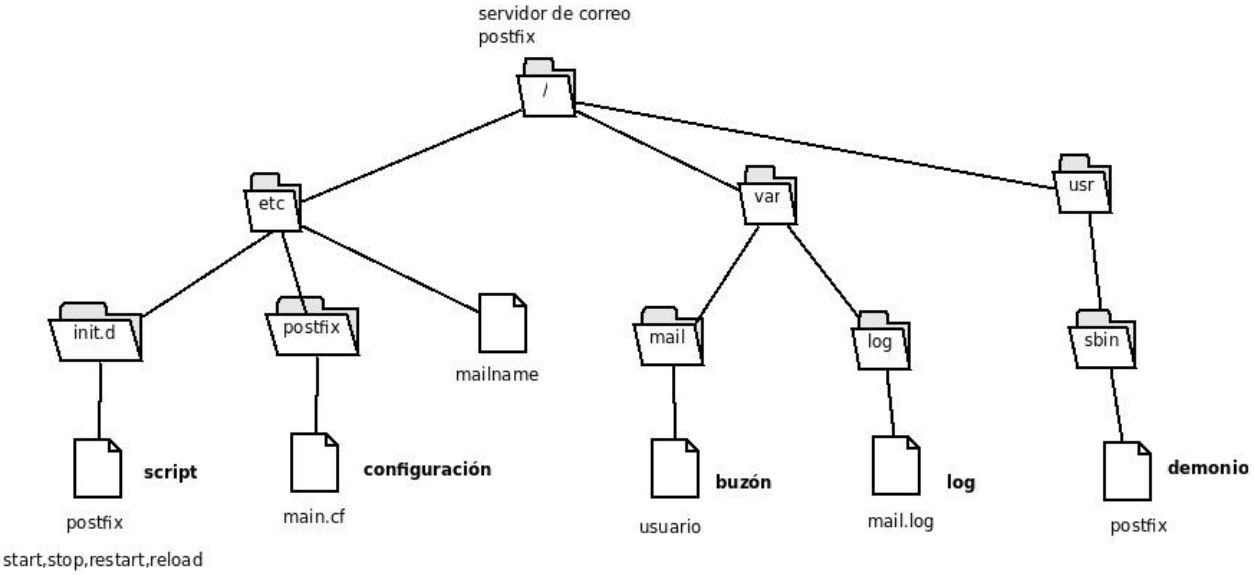
Directivas de configuración

Revisemos las directivas a tener en cuenta en nuestra configuración (`/etc/postfix/main.cf`):

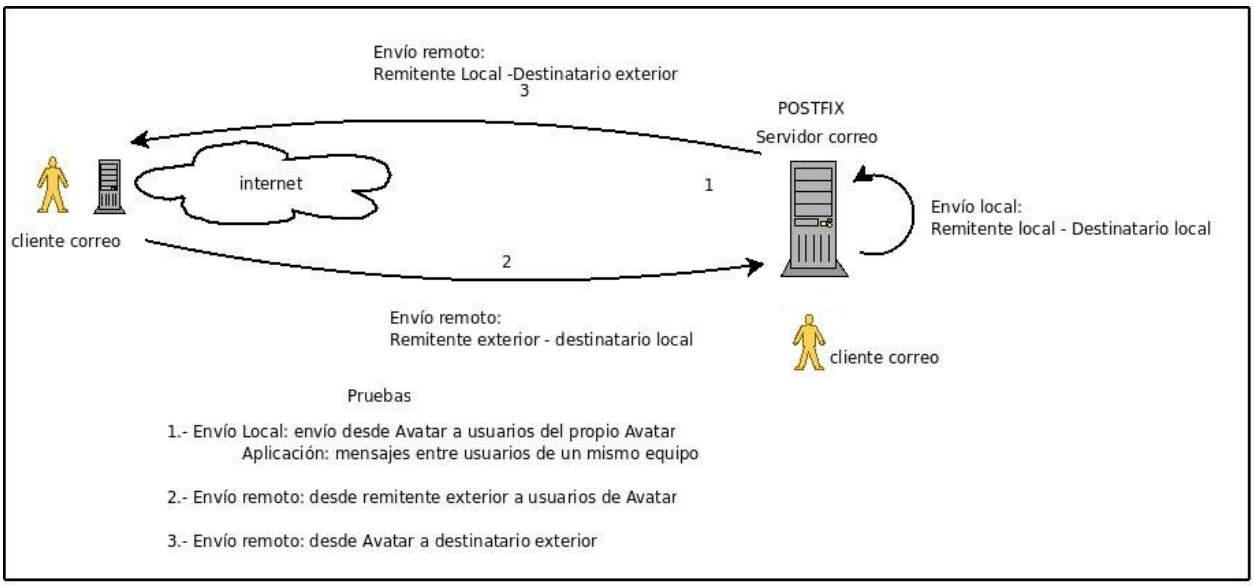
- `mydestination`: Observa que en la directiva `mydestination` se indican los dominios que serán propios del servidor de correo, es decir, el correo envíado a estos dominios está dirigirido a usuarios del propio servidor. Si el usuario existe, el mensaje será almacenado, sino el servidor devolverá un mensaje de error.
- `relay_domains`: Con la directiva `relay_domains` indicamos los dominios que serán reenviados. Por lo tanto se permitirán el envío de correos a usuarios de estos dominios.
- `mynetworks`: Con `mynetworks` se indican las IPs desde las que pueden enviarse mensajes.

- `myorigin`: Por último, con `myorigin` se indica el dominio con el que el servidor enviará correo, el cual está configurado en `/etc/mailname`.

Localización de los ficheros del servidor postfix



7.3 Gestión de correo desde el servidor



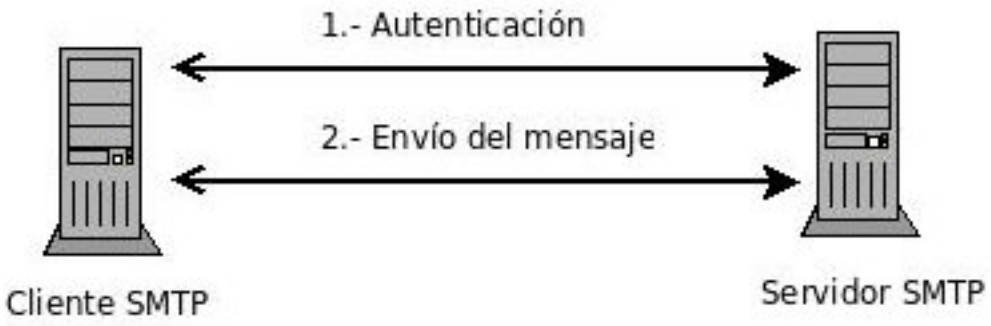
7.4 Caso 1: Envío local, entre usuarios del mismo servidor

Esta situación podemos aprovecharla para el envío de correos entre usuarios de la máquina, disponiendo de correo interno. Con utilidades como `mail usuario` podemos enviar mensajes a usuarios del propio sistema.

Vamos a realizar el envío utilizando una conexión `telnet`, muy interesante tanto para entender los pasos que sigue el protocolo SMTP:

```
root@vostro:/home/jose# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 mail2.josedomingo.org ESMTP Postfix (Debian/GNU)
HELO mail2.josedomingo.org
250 mail2.josedomingo.org
mail from: jose@josedomingo.org
250 2.1.0 Ok
rcpt to: usuario@josedomingo.org
250 2.1.5 Ok
data
354 End data with <CR><LF>.<CR><LF>
from: jose@josedomingo.org
to: usuario@josedomingo.org
subject: Prueba envio local

hola que tal

.
250 2.0.0 Ok: queued as DE3232C16A
quit
221 2.0.0 Bye
Connection closed by foreign host.
```

Pasemos a describir los pasos:



Fases en el envío de un mensaje de correo

- **Fase de autenticación** (puede haber otra si la sesión es cifrada). El cliente lanza los comandos siguientes para indicar qué usuario envía el correo, y a quién va dirigido. En nuestro ejemplo el cliente no llega a autenticarse al no estar configurado el servidor de correo para ello, tan sólo se intercambian estos mensajes.
  - EHLO o HELO "cadena presentándose el cliente ante el servidor"
  - MAIL FROM: "dirección de correo del remitente"
  - RCPT TO: "dirección de correo del destinatario"
- **Fase de envío del mensaje**. El cliente lanza la orden DATA y el servidor responde indicando "354 End data with .", lo que quiere decir que cuando el cliente finalice el mensaje y desee enviarlo debe escribir un punto y dar un retorno de carro, es decir, una vez a la tecla intro. El cliente lanza las cadenas
  - From: "dirección del remitente" + SALTO DE LÍNEA (ie intro)

- To: "dirección del destinatario" + SALTO DE LÍNEA (ie intro)
- Cc: "dirección del destinatario" para tener una copia + SALTO DE LÍNEA (ie intro)
- Bcc: "dirección del destinatario" para tener una copia ciega
- Date: "fecha" + SALTO DE LÍNEA (ie intro)
- Subject: "asunto" + SALTO DE LÍNEA (ie intro)
- MIME-Versión: "valor de la versión de MIME usada" + SALTO DE LÍNEA (ie intro)
- Otras cabeceras + SALTO DE LÍNEA (ie intro)
- DOS SALTOS DE LÍNEAS
- Escribe el mensaje en varias líneas
- DOS SALTOS DE LÍNEAS, y en el segundo escribe "." para finalizar el mensaje, y el cliente lo envía al servidor

Podemos comprobar el log `/var/log/mail.log` para comproar que se ha mandado el mansaje:

```
Feb 6 18:10:05 vostro postfix/smtpd[3660]: DE3232C16A: client=localhost[127.0.0.1]
Feb 6 18:11:07 vostro postfix/cleanup[3907]: DE3232C16A: message-id=<20120206171005.
↪DE3232C16A@mail2.josedomingo.org>
Feb 6 18:11:07 vostro postfix/qmgr[3531]: DE3232C16A: from=<jose@josedomingo.org>,␣
↪size=400, nrcpt=1 (queue active)
Feb 6 18:11:08 vostro postfix/local[3908]: DE3232C16A: to=<usuario@josedomingo.org>,␣
↪relay=local, delay=75, delays=74/0/0/1, dsn=2.0.0, status=sent (delivered to␣
↪command: procmail -a "$EXTENSION")
Feb 6 18:11:08 vostro postfix/qmgr[3531]: DE3232C16A: removed
Feb 6 18:11:09 vostro postfix/smtpd[3660]: disconnect from localhost[127.0.0.1]
```

Y podemos leer el mansaje del usuario «usuario» con el programa `mail`:

```
Mail version 8.1.2 01/15/2001. Type ? for help.
"/var/mail/usuario": 1 message 1 new
>N 1 jose@josedomingo. Mon Feb 6 18:11 15/548 Prueba envio local
```

7.5 Caso 2: Envío de correos desde internet a usuarios del servidor

DESDE EL AULA

Vamos a tener un correo de la forma `usuario@dominio_de_cada_alumno`, para nuestro ejemplos pongamos `jose@josedom.gonzalonazareno.org`.

Tenemos que tener en cuenta los siguientes aspectos:

1. Si queremos recibir correos desde intenet a nuestro servidor, todos nuestros de dominios tinen que apuntar a nuestra ip pública 80.59.1.152, sin embargo no hay que tocar el DNS de cdmon ya que tenemos un registro genérico que envía a 80.59.1.152 cualquier cosa de .gonzalonazareno.org que no tenga un registro tipo ADDRESS. Prueba a hacer un `dig loquesea.gonzalonazareno.org`.
2. Cuando se recibe un correo en esa dirección pública, lo recibe el servidor de correo que tenemos en babuino. Esto lo hace el cortafuegos de macaco.
3. Tenemos que configurar el servidor de correos de babuino para que haga relay con los correos cuyo destinos sean nuestros dominios, es decir el correo que vaya a `josedom.gonzalonazarno.org` lo tiene que enviar al servidor de correos de ese dominio, para ello:

- Añadimos en la directiva `realy_domains`, del servidor de correos de babuino, cada uno de los nombres de dominios a los que queremos reenviar los mensajes.
- Para que conozca la IP de nuestro servidor de correo tendremos que crear un registro MX en nuestro servidor DNS para realizar la resolución.

4. Con la configuración que tenemos en el servidor de correo de nuestra máquina debe ser suficiente para recibir el correo. Recuerda mandar un mensaje a un usuario que exista en el servidor.

DESDE CASA

Usando un nombre de dominio

En este caso vamos a utilizar un nombre de dominio, en mi caso he usado `josedomingo.org`. Tenemos que tener en cuenta los siguientes aspectos:

1. Configura el servidor postfix en tu ordenador, teniendo en cuenta que en el fichero `/etc/mailname` este tu nombre de dominio.
2. Configura tu DNS para que el registro MX apunte a un nombre de ordenador que está definido como un registro A a tu dirección IP pública (si es dinámica y por cualquier motivo te cambia tendrás que actualizar el registro). En mi caso el registro MX de josedomingo.org está configurado de este modo::

```
; <<>> DiG 9.7.3 <<>> -t mx josedomingo.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9147
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 4

;; QUESTION SECTION:
;josedomingo.org.        IN    MX

;; ANSWER SECTION:
josedomingo.org.    900    IN    MX    10 mail2.josedomingo.org.

;; AUTHORITY SECTION:
josedomingo.org.    21600    IN    NS    ns1.cdmon.net.
josedomingo.org.    21600    IN    NS    ns3.cdmon.net.
josedomingo.org.    21600    IN    NS    ns2.cdmon.net.

;; ADDITIONAL SECTION:
mail2.josedomingo.org.    900    IN    A    46.234.130.137
```

3. En tu router haz DNAT para redirigir las peticiones por el puerto 25 al ordenador local que tiene instalado el servidor de correos.

Usando un nombre de máquina

En el caso de que utilices un servicio gratuito como no-ip o dyndns, estarás reservando un nombre de máquina y no de dominio, por ejemplo `avatar.dyndns.com`, en este caso no tiene sentido configurar el registro MX:

1. En este caso el nombre de máquina (`avatar.dyndns.com`) por ejemplo, apuntará a tu dirección IP pública.
2. En tu router haz DNAT para redirigir las peticiones por el puerto 25 al ordenador local que tiene instalado el servidor de correos.

7.6 Alias y redirecciones

Alias

Cuando se define un alias para un determinado usuario se redirige el correo que llegue a otro usuario de la misma máquina. Los alias de correo se utilizan principalmente para gestionar el correo de las «cuentas de administración» y se definen en el fichero `/etc/aliases`, que tiene el siguiente aspecto::

```
# /etc/aliases
mailer-daemon: postmaster
postmaster: root
nobody: root
hostmaster: root
usenet: root
news: root
webmaster: root
www: root
ftp: root
abuse: root
noc: root
security: root
www-data: root
logcheck: root
root: alberto, jose, raul
```

En este caso el correo que llega a los usuarios postmaster, webmaster, etc. se redirige a la cuenta de root, que a su vez se redirige a los usuarios reales alberto, jose y raul, que son los administradores del equipo.

Cada vez que se modifica el fichero `/etc/aliases` hay que ejecutar la instrucción `newaliases` para que los cambios tengan efecto.

Redirecciones

Una redirección se utiliza para enviar el correo que llegue a un usuario a una cuenta de correo exterior. Para usuarios reales las redirecciones se definen en el fichero `~/.forward` y el formato de este fichero es simplemente un listado de cuentas de correo a las que se quiere redirigir el correo.

7.7 Caso 3: Envío de correo desde usuarios del servidor a correos de internet

DESDE EL AULA

En el caso de la configuración de nuestra red del insitituto sólo puede enviar correo el servidor de correo de babuino. Por lo tanto tenemos que configurar nuestro servidor para que utilice a babuino como relay para enviar nuestros correos, para ello, modificamos la siguiente directiva en el fichero de configuración:

```
ralayhost = babuino.gonzalonazareno.org
```

DESDE CASA

Con la configuración que tienes actualmente podrías mandar correos al exterior.

Si en tu casa tienes un dirección IP dinámica seguramente gmail /hotmail/yahoo lo rechaza, por estar en una lista de bloqueo, al intentar enviar un correo nos salen registro de este tipo:

```
postfix/pickup[6804]: 09B0634680: uid=1000 from=<alberto>
postfix/cleanup[6810]: 09B0634680:message-id=<20081231154700.09B0634680@m\
ortadelo>
postfix/qmgr[6802]: 09B0634680: from=<alberto@avatar.dynalias.com>, size=30\
7, nrcpt=1 (queue active)
postfix/smtp[6812]: 09B0634680: to=<una@hotmail.com>,relay=mx2.hotmail.co\
m[65.54.244.40]:25, delay=1.3, delays=0.03/0.04/0.92/0.3, dsn=5.0.0, stat\
us=bounced (host mx2.hotmail.com[65.54.244.40] said: 550 DY-001 Mail reje\
cted by Windows Live Hotmail for policy reasons. We generally do not acce\
pt email from dynamic IP's as they are not typically used to deliver unau\
thenticated SMTP e-mail to an Internet mail server. http://www.spamhaus.o\
rg maintains lists of dynamic and residential IP addresses. If you are no\
t an email/network admin please contact your E-mail/Internet Service Prov\
ider for help. Email/network admins, please visit http://postmaster.live.\
com for email delivery information and support (in reply to MAIL FROM com\
mand))
postfix/smtp[6812]: 09B0634680: lost connection with mx2.hotmail.com[65.5\
4.244.40] while sending RCPT TO
```

Para solucionar este problema tenemos varias soluciones, que veremos en los siguientes apartados:

- Enviar correos a partir de un realy host autentificado
- Estudiar las listas de bloqueos
- Crear registros SPF

7.8 Soluciones al problema del spam

Como hemos visto en el caso 3 es posible que nuestra ip dinámica que tenemos en casa sea rechazada por los servidores de correo. En este apartado vamos a ver distintas técnicas que nos permiten luchar contra el spam y que nos pueden permitir que nuestro correo llegue a su destino y no sea rechazado. Veamos las posibles técnicas:

- **SMTP submission**: Para que no se permita conectar directamente a un servidor de correo de destino, una técnica para evitar a los spammer sería cambiar el puerto estándar del protocolo SMTP, normalmente se cambia al 587. Para más información leer el siguiente artículo: Para que el correo llegue a buen puerto
- **Enviar correo a través Relay autentifcado**: En este caso podemos hacer relay utilizando un servidor de correos autentificado por ejemplo gmail: Configurar postfix a través de un relay host autenticado (Gmail)
- **SMTPd restrictions**: Podemos configurar nuestro servidor de correos para que filtre a los clientes que intentan usarlo por medio de algún parámetro del protocolo: HELO, MAIL FROM, RCPT TO,. . . , en el siguiente artículo puedes informarte sobre esta técnica y algunas otras que vamos a ver a continuación: SMTPd restrictions, SPF, DKIM and greylisting , Postfix restrictions.
- **Realtime blacklists (RBL)**: La técnica más importante para luchar contra el spam son las listas negras. Podemos configurar nuestro servidor de correo para que consulte en las distintas listas que existen la dirección de ip de origen How To Block Spam Before It Enters The Server (Postfix). Tenemos muchos sitios para ver si nuestra ip está en una lista negra:
  - mxtoolbox
  - Spamhaus Block List
  - Cisco IronPort SenderBase Security Network

- **Sender Policy Framework (SPF)**: Esta técnica consiste en publicar una serie de datos en un registro TXT del servidor DNS que haga que el servidor de correo donde llega el correo confíe en que el correo no es spam. Para más información lee el artículo: Sender Policy Framework (SPF)
- **DomainKeys Identified Mail (DKIM)**: Es un mecanismo que nos permite firmar un correo utilizando criptografía de clave pública. Para más información puede leer en la wikipedia: DomainKeys Identified Mail
- **Lita gris**: El servidor de correo si determina que el origen puede ser una amenaza, rechazada por defecto el correo, los servidores de correo legítimos volverán a enviar el correo transcurrido un tiempo, este correo será aceptado. Sin embargo un sistema que manda spam no suele reenviar el correo rechazado. Para más información puede leer en la wikipedia: Lista gris

Para terminar os dejo una presentación: Univ. de Deusto: Seguridad en sistemas de correo electrónico

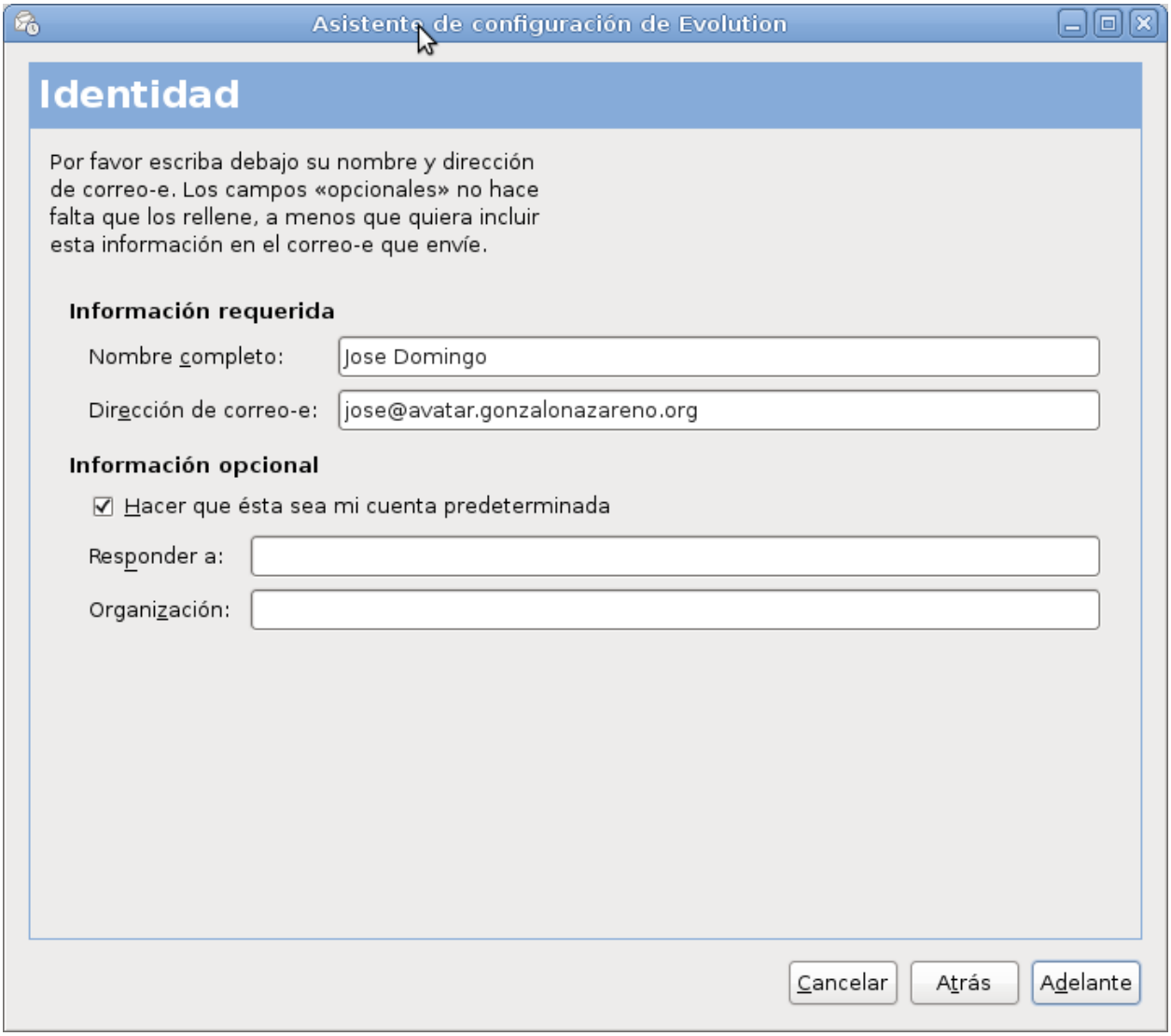
7.9 Caso 4: Envio de correo electrónico usando nuestro servidor de correos

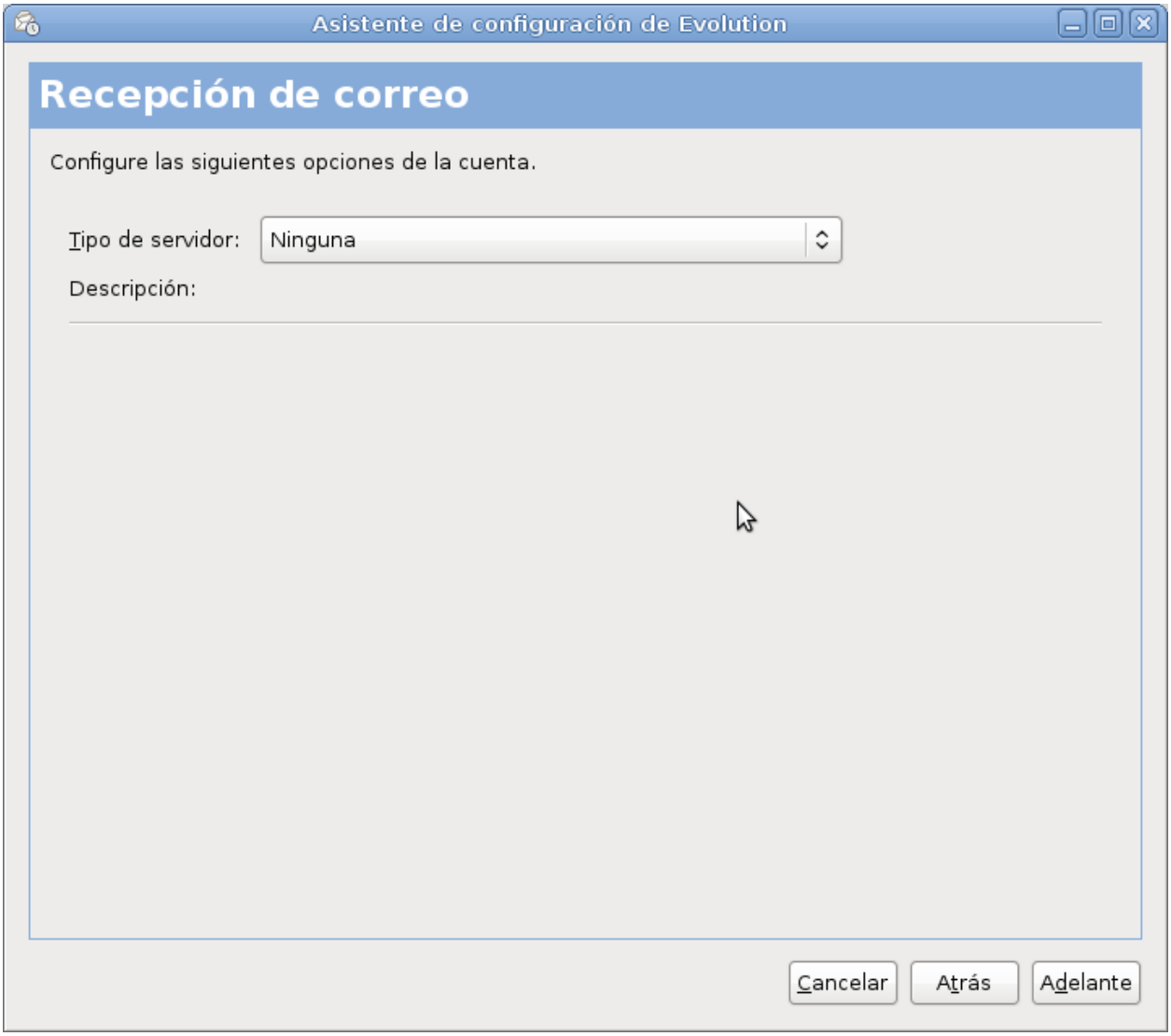
En este caso queremos que poder gestionar el envio y la recepción de corresos desde los ordenadores de neustra red local. Para ello debemos habilitar el envío de correo desde cliente de nuestra red. Para ello añade 192.168.1.0/24 (suponiendo que esta es nuestra red local) en la directiva mynetworks, quedando:

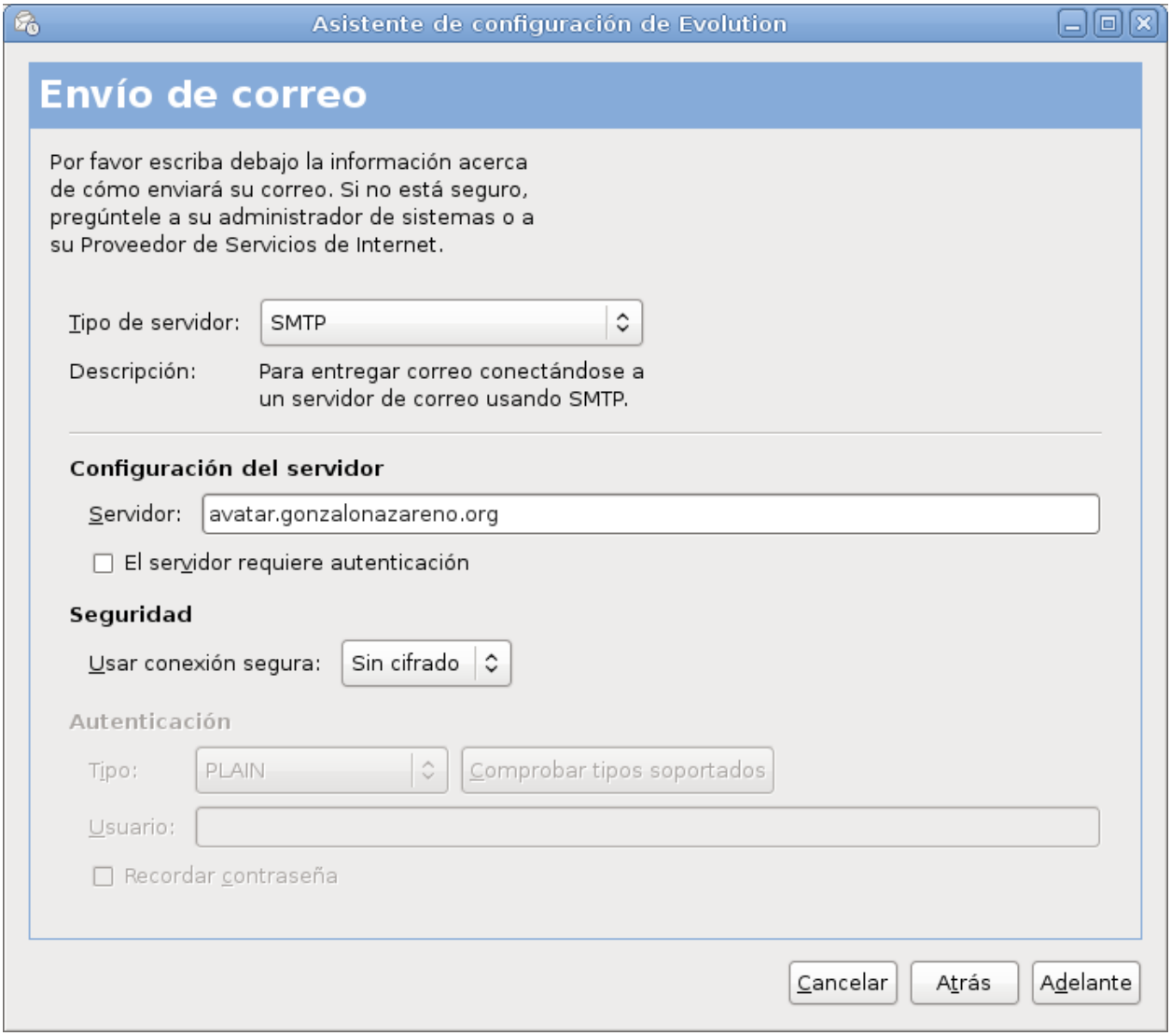
```
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::z1] 192.168.1.0/24
```

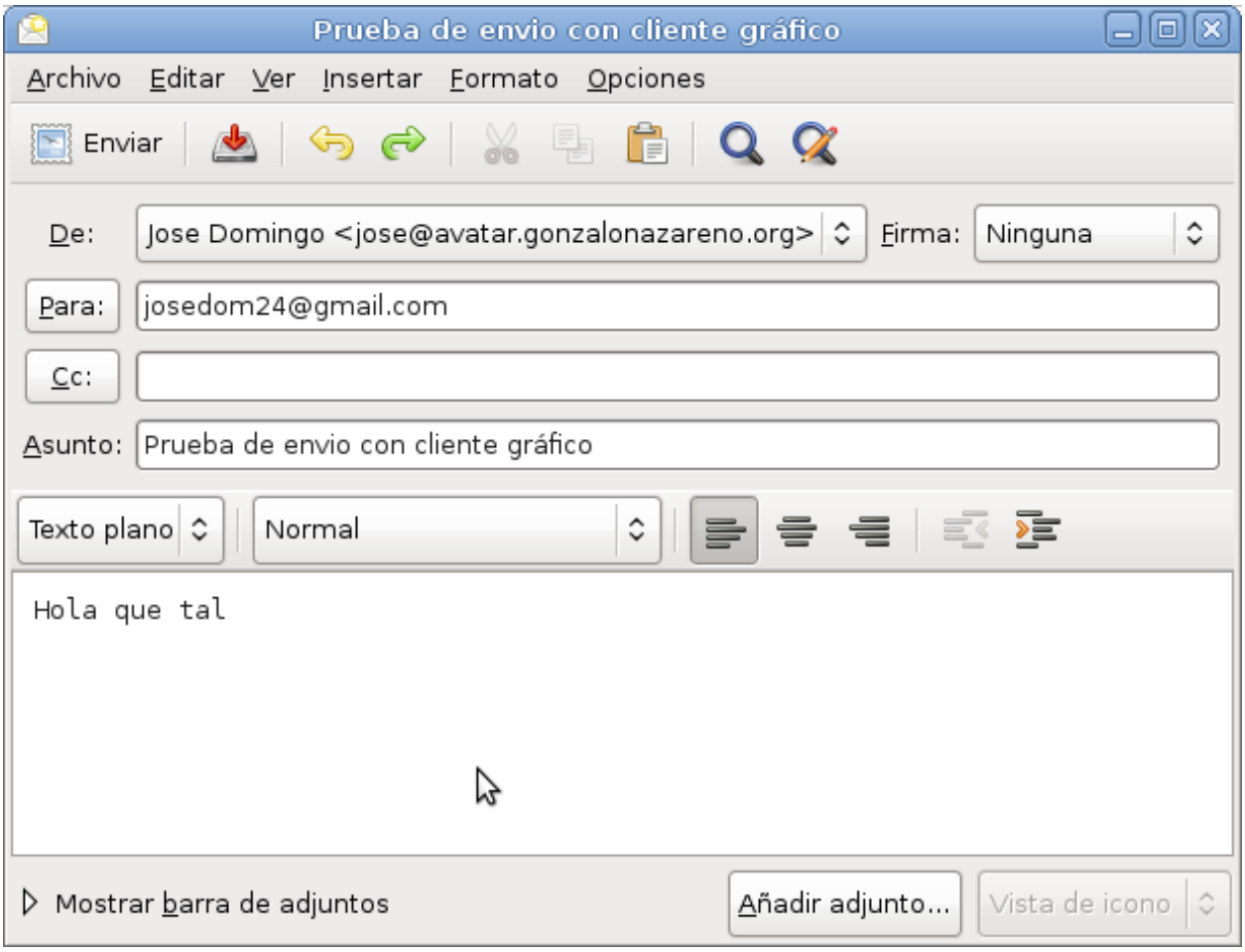
Es el momento de utilizar un cliente de correos (evolution, outlook, icedove, Thunderbird,...) en los clientes de nuestra red local para el envío de correos electrónicos. Para ello es necesario indicarle el nombre de nuestro servidor de correos, en mi caso `smtp.josedom.gonzalonazareno.org`, para ello desde los ordenadores de nuestra red local este nombre debe ser conocido, ya sea por resolución estática o usando un servidor DNS).

Veamos la configuración del cliente de correo Evolutión:


Asistente de configuración de Evolution
Identidad
Por favor escriba debajo su nombre y dirección de correo-e. Los campos «opcionales» no hace falta que los rellene, a menos que quiera incluir esta información en el correo-e que envíe.
Información requerida
Nombre completo:
Jose Domingo
Dirección de correo-e:
jose@avatar.gonzalonazareno.org
Información opcional
Hacer que ésta sea mi cuenta predeterminada
Responder a:
Organización:
Cancelar
Atrás
Adelante


Asistente de configuración de Evolution
Recepción de correo
Configure las siguientes opciones de la cuenta.
Tipo de servidor:
Ninguna
Descripción:
Cancelar
Atrás
Adelante


Asistente de configuración de Evolution
Envío de correo
Por favor escriba debajo la información acerca de cómo enviará su correo. Si no está seguro, pregúntele a su administrador de sistemas o a su Proveedor de Servicios de Internet.
Tipo de servidor:
SMTP
Descripción:
Para entregar correo conectándose a un servidor de correo usando SMTP.
Configuración del servidor
Servidor:
avatar.gonzalonazareno.org
El servidor requiere autenticación
Seguridad
Usar conexión segura:
Sin cifrado
Autenticación
Tipo:
PLAIN
Comprobar tipos soportados
Usuario:
Recordar contraseña
Cancelar
Atrás
Adelante



7.10 Caso 5: Recepción de correo electrónico usando nuestro servidor de correos

En este caso vamos a utilizar el protocolo pop3 y el protocolo imap para obtener los correos electrónicos que hemos recibio en nuestro servidor.

Ante de ellos tenemos que estudiar los ditintos tipos de buzones donde un servidor de correos puede guardar los mensajes:

- Buzón mbox: Todos los mensjaes están en un fichero, es el tipo de buzón que hemos utilizado hasta ahora.
- Buzón maildir: Los mensajes se guardan en una carpeta. Es imprescindible para que funcione el protocolo imap.

Configurando un bozón Maildir

Lo primero que vamos a hacer es configurar postfix para que guarde los correos en un buzón del tipo Maildir, para ello añadimos y modificamos las siguientes directivas de configuración en el fichero de configuración:

```
home_mailbox = Maildir/
mailbox_command =
```

Después de reinicar el servicio los correos se guardarán en un directorio Maildir que se creara en el directorio home de cada usuario.

7.10.1 Prorocolo pop3

En informática se utiliza el Post Office Protocol (POP3, Protocolo de la oficina de correo) en clientes locales de correo para obtener los mensajes de correo electrónico almacenados en un servidor remoto.

En nuestro caso el servidor pop3 que vamos a instalar se llama dovecot-pop3, para instalarlo, simplemente:

```
apt-get install dovecot-pop3d
```

La configuración que tenemos que realizar es la siguiente: cambiamos en `/etc/dovecot/conf.d/10-auth.conf`:

```
#disable_plaintext_auth = yes    ->    disable_plaintext_auth = no
```

Para que este habilitada la autentificación con contraseña en claro.

Cambiamos en el fichero `/etc/dovecot/conf.d/10-mail.conf`, donde se encuentra el buzón:

```
#mail_location = mbox:~/mail:INBOX=/var/mail/%u
mail_location = maildir:~/Maildir
```

Configuración del cliente de correo

Es la hora de configurar un cliente de coreo desde nuestro cliente, en este ejemplo vamos a ver la configuración en evolution. Además tenemos que tener creados en el DNS de nuestro servidor los nombres `pop3.josedom.gonzalonazareno.org`.

7.10.2 Protocolo imap

Internet Message Access Protocol, o su acrónimo IMAP, es un protocolo de red de acceso a mensajes electrónicos almacenados en un servidor. Mediante IMAP se puede tener acceso al correo electrónico desde cualquier equipo que tenga una conexión a Internet. IMAP tiene varias ventajas sobre POP, que es el otro protocolo empleado para obtener correo desde un servidor. Por ejemplo, es posible especificar en IMAP carpetas del lado servidor. Por otro lado, es más complejo que POP ya que permite visualizar los mensajes de manera remota y no descargando los mensajes como lo hace POP.

Para instalar el servidor IMAP ejecutamos la siguiente instrucción:

```
apt-get install dovecot-imapd
```

Ya podríamos configurar nuestro cliente de correos para recibir correo utilizando el protocolo imap, para ello tendríamos creado un nombre en el servidor DNS, por ejemplo, `imap.josedom.gonzalonazreno.org`.

Nota: Podrías hacer la prueba de ver la diferencia entre los dos protocolo: como si usas pop3 los correos se borran del servidor (hay que configurar algunos clientes para que funcionen de esta forma) y cómo sin embargo al usar el servidor imap los correos no se borran del servidor.

En este caso también puedes instalar un cliente web de correos para leer los correos del servidor. Hay muchas: squirrelmail, round cube,...

7.11 Antispam y antivirus en nuestro ervidor de correos

En este apartado vamos a realizar una introducción a la configuración de nuestro servidor de correos postfix para que sea capaz de determinar si los correos que le llegan tienen algun virus o son spam. Para ello vamos a utilizar tres

programas:

- amavisd: Es una interfaz entre el servidor de correos y el antivirus y antispam.
- ClamAV: Un antivirus.
- SpamAssassin: Un software para detectar el spam.

Es esquema de funcionamiento es el siguiente::

```
                    [SpamAssassin]
                                          ^
                                          |
Email --> [(Port 25) Postfix] --> [(10024) amavisd-new] --> [(10025) Postfix] -->␣
↪Mailbox
                                          |
                                          v
                                      [ClamAV]
```

Os voy a dejar los pasos para realizar la instalación, este manual nace de mi experiencia, si veis algún cambio o hay algo incorrecto, por favor decírmelo. Empezamos con la instalación (vamos a instalar los todos los paquetes necesarios)::

```
apt-get install amavisd-new spamassassin clamav clamav-daemon
```

Instalamos descompresores y utilidades::

```
apt-get install unrar-free zoo unzip bzip2 libnet-ph-perl libnet-snpp-perl libnet-
↪telnet-perl nomarch lzop
```

7.11.1 amavisd

En el fichero `/etc/amavis/conf.d/15-content_filter_mode` descomentamos las siguientes líneas si vamos a activar el antivirus::

```
@bypass_virus_checks_maps = (
    \%bypass_virus_checks, \@bypass_virus_checks_acl, \$bypass_virus_checks_re);
```

Y las siguientes si vamos a activar el antispam::

```
@bypass_spam_checks_maps = (
    \%bypass_spam_checks, \@bypass_spam_checks_acl, \$bypass_spam_checks_re);
```

En el fichero `etc/amavis/conf.d/20-debian_defaults`, configuramos que hacer con los correos sospechosos, la configuración que viene por defecto es correcta pero podemos hacer algunos cambios, por ejemplo podemos dejar pasar el spam, aunque lo marquemos::

```
$final_spam_destiny = D_PASS;
```

Por último::

```
adduser clamav amavis
systemctl restart amavis
```

Para que amavis pueda comunicar con postfix, añadimos a la configuración las siguientes líneas::

```
postconf -e 'content_filter = amavis:[127.0.0.1]:10024'
postconf -e 'receive_override_options = no_address_mappings'
```

Añadimos las siguientes líneas al fichero `/etc/postfix/master.cf`::

```
amavis unix - - - - 2 smtp
        -o smtp_data_done_timeout=1200
        -o smtp_send_xforward_command=yes

127.0.0.1:10025 inet n - - - - smtpd
        -o content_filter=
        -o local_recipient_maps=
        -o relay_recipient_maps=
        -o smtpd_restriction_classes=
        -o smtpd_client_restrictions=
        -o smtpd_helo_restrictions=
        -o smtpd_sender_restrictions=
        -o smtpd_recipient_restrictions=permit_mynetworks,reject
        -o mynetworks=127.0.0.0/8
        -o strict_rfc821_envelopes=yes
        -o receive_override_options=no_unknown_recipient_checks,no_header_body_checks
        -o smtpd_bind_address=127.0.0.1
```

Y podemos ver los puertos por los que se está escuchando::

```
tcp        0      0 127.0.0.1:10024         0.0.0.0:*               LISTEN      18955/
↪amavisd-new (
tcp        0      0 127.0.0.1:10025         0.0.0.0:*               LISTEN      19134/
↪master
tcp        0      0 127.0.0.1:783           0.0.0.0:*               LISTEN      16089/
↪spamassassin.
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN      19134/
↪master
```

7.11.2 clamav

Actualizamos el antivirus::

```
freshclam
systemctl restart clamav-daemon
```

7.11.3 spamassassin

Lo activamos en el fichero `/etc/default/spamassassin`::

```
ENABLED=1
```

El fichero de configuración es `/etc/spamassassin/local.cf`

Para más información puedes leer los siguientes artículos:

- howtoforge - Integrating amavisd-new Into Postfix For Spam- And Virus-Scanning
- Correo SPAM
- Como probar la efectividad y eficiencia del programa antivirus instalado
- Los test de spamassassin

CAPÍTULO 8

Servidor proxy/cache Squid

Indice

8.1 Enlaces interesantes

- Presentación: Squid
- Proxy-cache Squid
- Squid, un proxy caché para GNU/Linux
- Manual de Instalación de Servidor Proxy Web con Squid

8.2 Ejercicio: Instalación y configuración básica de squid

1. Instala el proxy-cache squid3 y comprueba la configuración básica del servidor:
   - Comprueba el puerto por el que está escuchando el servidor.
   - Asigna como memoría de la cache 300 Mb.
   - Define un tamaño de cache de 8 Gb.
   - Define el máximo tamaño de los elementos cacheados a 20 Mb.
   - Comprueba los ficheros de log y con que formáto se guarda la información.
   - Modifica la configuración para que los errores salgan en español.
2. Una vez realizada la configuración básica, reinicia el servicio. Configura de manera manual un navegador y comienza a navegar utilizando el proxy.

   ¿Funciona? No, el servidor viene configurado por defecto para que sólo se puedan realizar conexiones desde localhost.
3. Para solucionar esto y poder acceder desde nuetra LAN tenemos que modificar una regla de acceso ACL:

- Descomentamos una regla ACL localnet y la ponemos de la siguiente manera::

  acl localnet src 10.0.0.0/24

- Descomentamos la siguiente línea que nos permite el acceso a la red que hemos definido anteriormente::

  http_access allow localnet

Reinicamos, y comprobamos: miramos el fichero `/var/log/squid3/access.log`, y vemos las peticiones que se han guardado en la cache. Puedes ecntrar información de cómo leer los logs en las siguientes direcciones:

- linofee.org - Squid access.log
- Analizar los logs de access.log de squid

8.3 Gestionando la cache de Squid (parte 1)

En este apartado vamos a estudiar cómo funciona la caché de squid:

- Gestión de la cache por mecanismos de validación
- Gestión de la cache por mecanismos de frescura
- Como evitar el cacheo de nuestro contenido

Gestionando la caché de Squid (parte 1)

Para obtener más información puedes ver los siguientes enlaces:

- Tutorial Cache Web: cómo gestionar el cacheo de nuestros contenidos
- Two Important Differences between Firefox and IE Caching
- La memoria cache de Firefox: fallo o característica
- REDbot.org
- Tutorial Cache Web: cómo gestionar el cacheo de nuestros contenidos
- Optimización: Agregar Headers de Expiración
- Reduciendo el tráfico usando cache en Apache

8.4 Ejercicio: Controlando la cache de squid

Partimos de la siguiente situación: tenemos instalado squid3 en un servidor, un navegador en la máquina real está configurado para usar el proxy (además en este navegador hemos desactivado la función de cache) y vamos acceder desde este navegador a una página html almacenada en un servidor web apache2 instalado en otro servidor.

Gestión de la cache por mecanismos de validación

En este primer punto vamos a acceder a la página html, y sólo va a entrar en juego el parámetro de cabacera `Last-Modified`. Una vez cacheada la página, si volvemos acceder a ella se preguntará al servidor si se ha modificado, el servidor responderá con la cabecera HTTP, y si la copia que tenemos es nueva se servirá directamente.

- En este caso al refrescar la página con F5 nos vamos encontrando en el fichero access.log con información del tipo `TCP_MEM_HIT` o `TCP_HIT`, es decir acierto en la cache.
- Si modificamos la página, el servidor cambiará el parámetro `Last-modified` y por tanto la copia que tenemos almacendas ya no sera válida, por lo que nos bajaremos del servidor la página modificada y la volveremos almacenar. En este caso nos encontraremos en el fichero access.log una línea del tipo `TCP_REFRESH_MODIFIED`, indicando que la página accedida ha sido modificada.

- Si simulamos que se ha perdido la conexión con el servidor, parando el servicio apache2, aunque se intenta verificar con el servidor si la página ha sido modificada (`TCP_REFRESH_FAIL`), se seguirá sirviendo la copia que tenemos.

Gestión de la cache por mecanismos de frescura

`Expires`

El parámetro `Expires` de la cabecera de un mensaje HTTP indica cuando o cada cuanto tiempo la página guardad en cache no es válida y por lo tanto hay que bajarse otra del servidor. Para cambiar este parámetro de la cabecera vamos a usar el `mod_expire` de Apache2. Para ello nos aseguramos que está activo::

```
a2enmod expires
```

Y modificamos el fichero de configuración default de la siguiente manera::

```
<Directory /var/www/>
ExpiresActive On
ExpiresDefault "access plus 3 seconds"
...
```

En este caso estamos diciendo que todos los ficheros alojados en `/var/www` tienen una caducida de 3 segundos. Cada tres segundos se va a obligar a descargarse la nueva copia del servidor.

Puedes comprobar el valor del parámetro usando el comando `HEAD`.

Si vamos recargando la página con F5 nos daremos cuenta que cada 3 segundos se produce un `TCP_REFRESH_UNMODIFIED`, es decir se obliga a la descarga de la página, durante el tiempo intermedio se supone que la página es válida.

Como evitar el cacheo de nuestro contenido

Como estudiado en la teoría el contenido del parámetro `Control-cache` de la cabecera de un mensaje HTTP indica al proxy-cache que puede hacer con dicho contenido.

Para poder modificar este parámtetro vamos a usar el `mod_headers` de apache2, para ello nos aseguramos que este activo::

```
a2enmod headers
```

A continuación podemos modifcar el fichero de configuración default y añadir las siguiente líneas para evitar que se pueda cachear nuestro docuemento html::

```
<Directory /var/www/>
Header set Cache-Control "private, no-cache, no-store"
Header set Pragma "no-cache"
...
```

En este caso cuando vamos refrescando nuestra página en el navegadorobtenemos un mensaje `TCP_MISS` en el fichero `access.log` indicando que nuestro documento no ha sido almacenado.

8.5 Gestionando la cache de Squid (parte 2)

En este apartado vamos a estudiar las posibilidades que ofrece squid a los administradores para forzar el «cacheo» de determinados elementos, incluso ignorando las cabeceras de control de caché.

- Gestionando la caché de Squid (parte 2)

Para obtener más información puedes ver los siguientes enlaces:

- refresh_pattern, controlando la cache

8.6 Gestionando el proxy de Squid

En este apartado vamos a ver como controlar el proxy por medio de ACL.

Para crear controles de acceso basados en el **origen** de la petición del cliente podemos usar los siguientes tipos de elementos de ACL:

- `src`: Dirección IP del cliente, puede ser una sola dirección lista o rango de direcciones IP, soporta el uso de mascaras de subred en formato CIDR
- `proxy_auth`: Autenticación de usuarios vía procesos externos
- `browser`: User-agent del navegador web que realiza la petición

Para crear controles de acceso basados en el **destino** de la petición encontramos los siguientes tipos de elementos de ACL:

- `dstdomain`: Este tipo define uno o más dominios destino solicitados por el cliente
- `url_regex`: Tipo con soporte de expresiones regulares para el URL solicitado por el cliente
- `urlpath_regex`: URL-path regular expression pattern matching, leaves out the protocol and hostname
- `port`: Este tipo define uno o más números de puerto destino solicitados por el cliente
- `method`: Este tipo define el método usado por el cliente para la petición HTTP (get, post, etc)

Para crear controles de acceso basados en el **tipo MIME** de la solicitud o respuesta de la petición podemos usar los siguientes tipos de elementos de ACL:

- `req_mime_type`: regular expression pattern matching on the request content-type header
- `rep_mime_type`: regular expression pattern matching on the reply (downloaded content) content-type header.

Además es posible crear controles de acceso basados en el **tiempo** en el que se realiza la petición, y usando procesos externos, por ejemplo para realizar autenticación basada en grupos, las ACLs que podemos usar son:

- `time`: hora del día, y día de la semana

8.6.1 Reglas de control de acceso

Los controles de acceso se realizan principalmente con la directiva http_access para peticiones HTTP y http_reply_access para las respuestas HTTP.

Las reglas `http_access` se usan para permitir o denegar el acceso a uno o más elementos de ACL, es decir, se podría evaluar tanto el origen: dirección IP, usuario, o el destino: dominio o URL de la petición, por mencionar algunos tipos. Squid tomará toda la información posible de las cabeceras de la petición HTTP.

El esquema más simple las reglas http_access para determinar el acceso a un a un elemento sería el siguiente::

```
http_access allow|deny acl
```

Squid evalúa las reglas en el orden en el que son escritas, es decir, de arriba hacía abajo. Si la primer regla no hace coincidencia con la petición, entonces el squid realizará una operación de tipo OR y evaluará los elementos de la siguiente regla de acceso::

```
http_access allow|deny acl
                OR
http_access allow|deny acl
                OR
...
```

Si en una regla de acceso hay más de un elemento de ACL, el sistema utiliza el operador AND para cada elemento de la regla, esto quiere decir, que todos los elementos de la ACL deben hacer coincidencia para que una acción se aplique::

```
http_access allow|deny acl AND acl AND ...
                OR
http_access allow|deny acl AND acl AND ...
                OR
...
http_access deny all
```

Para más información sobre la creación de ACL puedes mirar los siguientes enlaces:

- Introducción a los esquemas de control de acceso en Squid
- Manual de Instalación de Servidor Proxy Web con Squid

8.6.2 Autenticación por usuarios y grupos con Squid

En esta sección se describen los procedimientos para configurar el proxy Squid para autenticar usuarios usando diferentes métidos de autenticación.

Módulos de autenticación de usuarios Squid

- NCSA Usa un archivo de usuarios y contraseñas al estilo NCSA
- LDAP Usa el protocolo Lightweight Directory Access Protocol
- MSNT Usa un dominio de autenticación Windows NT
- PAM Usa los módulos de autenticación PAM
- SMB Usa un servidor SMB como Windows NT o Samba

Para autentificar usuarios hay que usar ACL de tipo `proxy_auth`.

Para más información:

- Introducción a los esquemas de control de acceso en Squid

8.7 Ejercicio: Configuración de los esquemas de control de acceso

Crea diferentes ACL para obtener los siguientes resultados:

Reglas de acceso basadas en direcciones IP

1. Sólo permitir el acceso a squid desde la 10.0.0.2.
2. Sólo permitir el acceso en un rango de direcciones (10.0.0.2-10.0.0.4)
3. Permitir el acceso a la red 10.0.0.0
4. No permitir el acceso a las direcciones IP que tenemos listadas en el fichero ip_sin_internet.txt

5. Permitir el acceso a la red 10.0.0.0, excluyendo al rango 10.0.0.10-10.0.0.20 Reglas de acceso basadas en puertos, dominios, URLs y tipos MIME
6. Comprueba el acl que viene en el fichero de configuración donde se indican los puertos permitidos.
7. Explica que conseguimos con las siguientes ACL::

```
acl Safe_ports port 80 21 443 563 70 210 1025-65535
acl SSL_ports port 443
acl CONNECT method CONNECT
...
...
...
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
```

8. Evita que nos podamos conectar al dominio youtube.com y todos sus subdominios.
9. Evita que nos podamos conectar a los dominios listados en el fichero url_prohibidas.txt (lista negra)
10. Permite sólo la navegación a un conjuntos de nombres de dominios. (lista blanca)
11. Deniega el acceso a toda URL que contenga la palabra "informatica" (no tener en cuenta mayusculas y minusculas)
12. Deniega el acceso a toda URL que contenga las plabras "betis" o "sevilla"
13. Utilizando las ACL de tipo urlpath_regex, impide el acceso a ficheros pdf.
14. Usando el tipo MIME: evita el acceso a los ficheros de CSS (text/css), a los jpg (image/jpeg)
15. Evita la visualización de los contenidos flash

    Otras reglas de acceso

16. Evita que se pueda navegar con el navgador Internet Explorer
17. Permite la navegación sólo los días entre semana de 8:00 de la mañana a las 14:00 de la tarde.

8.8 Ejercicio: Autenticación básica por usuarios y grupos con Squid

Configura squid para controlar el acceso de distintos tipos de usuarios:

- Los usarios del perfil "alumnos" solo pueden navegar a páginas cuya URL contenga gonzalonazareno.
- Los usarios del perfil "profesores" admeás pueden navegar por paǵinas que contengan la palabra "juntadeandalucia".
- Los usarios del perfil "equipo directivo" pueden navegar por cualquier página.

8.9 Ejercicio: Configuración de parámetros de proxy en clientes web

Configuración manual de parámetros de proxy

Ya hemos estudiado cómo se configuran los navegadores web gráficos. En este ejercicio configura un equipo (sin entorno gráfico) para que al navegar con lynx se este usando el proxy.

Configuración de parámetros de proxy usando script Proxy Auto-config

Configura el servidor web apache2 para que sirva un fichero proxy.pac. Crea dicho fichero con la configuración básica del proxy, e indica las siguientes exclusiones: no se usa el proxy cuando se accede al dominio gonzalonazareno.org y no se usa el proxy cuando se accede a localhost.

Configura el cliente web en la opción: Configuración automática de proxy

Configuración de parámetros de proxy usando la detección automática WPAD

1. Configura un servidor dhcp que permita que los clientes al recibir la configuración dinámica configuren los parámtros de acceso al proxy.
2. Configura el servidor bind9 que permita a los clientes con direccionamiento estático encontrar la configuración del proxy.

Puedes encontrar mucha más información en: FindProxyForUrl

Proxy transparente

En el esquema de red de clase::

```
iptables -t nat -A PREROUTING ! -s 10.0.0.1 -i virbr1 -p tcp --dport 80 -j DNAT --to␣
↪10.0.0.1:3128
```

En el esquema router+nat+proxy::

```
iptables -t nat -A PREROUTING  -s 10.0.0.0/24 -i virbr1 -p tcp --dport 80 -j REDIRECT␣
↪--to-port 3128
```

8.10 Otras herramientas: Dansguardians y Sarg

8.10.1 Dansguardians

DansGuardian es un software de filtro de contenido, diseñado para controlar el acceso a sitios web.

- Página oficial de dansguardian
- Filtrando contenido (squid+dansguardian+iptables)
- urlblacklist.com

8.10.2 Sarg

Es una herramienta para generar informes de acceso a un proxy

- Página oficial de Sarge
- Instación y configuración de Sarg en Debian Jessie

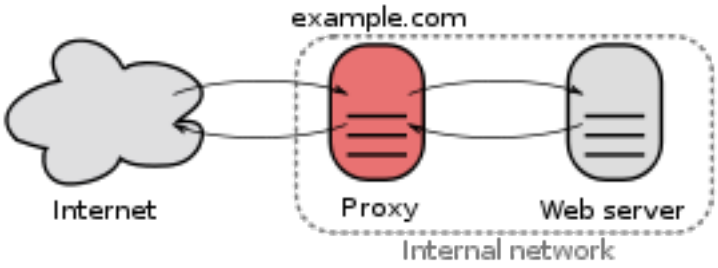
CAPÍTULO 9

Proxy inverso y Balanceador de carga

9.1 Proxy inverso

9.1.1 Configuración de un proxy inverso

Un proxy inverso es un tipo de servidor proxy que recupera recursos en nombre de un cliente desde uno o más servidores. Por lo tanto el cliente hace la petición al puerto 80 del proxy, y éste es el que hace la petición al servidor web que normalmente está en una red interna no accesible desde el cliente.



Apache como proxy inverso

Apache2.4 puede funcionar como proxy inverso usando el módulo `proxy` junto a otros módulos, por ejemplo:

- proxy_http: Para trabajar con el protocolo HTTP.
- proxy_ftp: Para trabajar con el protocolo FTP.
- proxy_html: Permite reescribir los enlaces HTML en el espacio de direcciones de un proxy.
- proxy_ajp: Para trabajar con el protocolo AJP para Tomcat.
- ...

Por lo tanto, para empezar, vamos activar los módulos que necesitamos:

```
# a2enmod proxy proxy_http
```

Ejemplo de utilización de proxy inverso

Tenemos a nuestra disposición un servidor interno (no accesible desde el cliente) en la dirección privada, con el nombre de `interno.example.org`. Tenemos un servidor que va a funcionar de proxy, llamado `proxy.example.org` con dos interfaces de red: una pública conectada a la red donde se encuentra el cliente, y otra interna conectada a la red donde se encuentra el servidor interno.

Sirviendo una página estática

En nuestro servidor interno hemos creado un virtual host para servir una página estática, `index.html`. Vamos a utilizar la directiva `ProvyPass` en el fichero de configuración del virtual host, de la siguiente forma:

```
ProxyPass "/web/" "http://interno.example.org/"
```

También lo podemos configurar de forma similar con:

```
<Location "/web/">
    ProxyPass "http://interno.example.org/"
</Location>
```

Evidentemente debe funcionar la resolución de nombre para que el proxy pueda acceder al servidor interno.

De esta manera al acceder desde el cliente la URL `http://proxy.example.org/web/` se mostraría la página que se encuentra en el servidor interno.

El probelma de las redirecciones

Cuando creamos una redirección en un servidor web y el cliente intenta acceder al recurso, el servidor manda una respuesta con código de estado `301` o `302`, e indica la URL de la nueva ubicación del recurso en una cabecera HTTP llamada `Location`.

Si hemos configurado una redirección en el servidor interno, cuando se accede al recurso a través del proxy, la redirección se realiza pero la cabecera `Location` viene referencia la dirección del servidor interno, por lo que el cliente es incapaz de acceder a la nueva ubicación. Para solucionarlo utilizamos la directiva `ProxyPassReverse` que se encarga de reescribir la URL de la cabecera `Location`.

La configuración quedaría:

```
ProxyPass "/web/" "http://interno.example.org/"
ProxyPassReverse "/web/" "http://interno.example.org/"
```

O de esta otra forma:

```
<Location "/web/">
    ProxyPass "http://interno.example.org/"
    ProxyPassReverse "http://interno.example.org/"
</Location>
```

El problema de las rutas HTML

La página que servimos a través del proxy que se guarda en el servidor interno puede tener declarada rutas, por ejemplo en imágenes o enlaces. Nos podemos encontrar con diferentes tipos de rutas:

- `http://interno.example.org/imagen.jpg`: Una ruta absoluta donde aparece la dirección del servidor interno y que evidentemente el cliente no va a poder seguir.
- `/imagen.jpg`: Una ruta absoluta, referenciada a la raíz del `DocumentRoot`.
- `imagen.jpg`: Una ruta relativa.

Si tenemos una ruta relativa, el cliente la va a poder seguir sin problema cuando accede a través del proxy, pero si tenemos una ruta como la segunda no lo va a poder hacer, porque en el `DocumentRoot` del proxy no existe este recurso.

Para solucionar este problema debemos reescribir el HTML para cambiar la referencia del enlace. Para ello necesitamos activar un nuevo módulo:

```
# a2enmod proxy_html
```

Y realizar la siguiente configuración:

```
ProxyPass "/web/"  "http://interno.example.org/"
ProxyPassReverse "/web/"  "http://interno.example.org/"
ProxyHTMLURLMap http://interno.example.org /web
<Location /web/>
    ProxyPassReverse /
    ProxyHTMLEnable On
    ProxyHTMLURLMap / /web/
</Location>
```

Como vemos hemos configurado un proxy para HTML, que será responsable de reescribir todos las rutas que contiene el HYML, utilizando la directiva `ProxyHTMLURLMap`:

```
ProxyHTMLURLMap http://interno.example.org /web
```

Es importante no poner la barra final, cuando se encuentra una ruta que coincide con el primer patrón se reescribe con el segundo, esta regla reescribe las ruta del tipo de la primera opción que hemos visto anteriormente. Para arreglar la rutas de la segunda opción, utilizamos dentro de la sección `Location`:

```
ProxyHTMLURLMap / /web/
```

Después de iniciar comprobamos que al intentar acceder al proxy obtenemos un error en el navegador del cliente «Error de codificación de contenido».

Sirviendo contenido multimedia

Acabamos de configurar un proxy que examina y reescribe el HTML de nuestro sitio web, pero evidentemente existe más contenido en nuestro sitio que no es HTML y no debería ser procesado por `proxy_html`. Esto se soluciona verificando la cabecera del contenido y rechazando todos los contenidos que no tengan el tipo MIME adecuado.

Pero tenemos un problema: normalmente se comprime el contenido HTML, y encontramos cabeceras de este tipo:

```
Content-Type: text/html
Content-Encoding: gzip
```

Este contenido no debería pasar por el analizador de `proxy_html`. Para solucionar esto podemos negarnos a admitir la compresión. La eliminación de cualquier cabecera de petición `Accept-Encoding` hace el trabajo. Para ello podemos utilizar la directiva `RequestHeader` del módulos `headers`, por lo tanto activamos el módulo:

```
# a2enmod headers
```

Y usamos la directiva `RequestHeader` dentro del la sección `Location`:

```
ProxyPass "/web/"  "http://interno.example.org/"
ProxyPassReverse "/web/"  "http://interno.example.org/"
ProxyHTMLURLMap http://interno.example.org /web
<Location /web/>
    ProxyPassReverse /
    ProxyHTMLEnable On
    ProxyHTMLURLMap / /web/
    RequestHeader unset Accept-Encoding
</Location>
```

Ahora si podemos acceder a la página completa a través del proxy.

Sirviendo contenido con HTTPS

Una situación similar surge en el caso del contenido encriptado (https). En este caso, usando el módulo `ssl` y un certificado en el proxy, de modo que la sesión segura real se encuentre entre el navegador y el proxy, no al servidor interno.

9.1.2 Enlaces interesantes proxy inverso

- Servidor proxy inverso: componente central de la seguridad
- apache2:Reverse Proxy Guide
- apache2: Caching Guide
- nginx: Module ngx_http_proxy_module
- Creating a caching proxy server with apache
- How To Use Apache as a Reverse Proxy with mod_proxy on Ubuntu 16.04
- Understanding Nginx HTTP Proxying, Load Balancing, Buffering, and Caching
- How To Configure Nginx as a Reverse Proxy for Apache
- Varnish with multiple sites and multiple IPs

Configuración del proxy visto en clase

```
ProxyPass "/ejercicio1/"  "http://ejercicio.gonzalonazareno.org/"
ProxyPassReverse "/ejercicio1/"  "http://ejercicio.gonzalonazareno.org/"
ProxyHTMLURLMap http://ejercicio.gonzalonazareno.org /ejercicio1
<Location /ejercicio1/>
    ProxyPassReverse /
    ProxyHTMLEnable On
    ProxyHTMLURLMap / /ejercicio1/
    RequestHeader    unset  Accept-Encoding
</Location>
```

Referencia: http://www.apachetutor.org/admin/reverseproxies

9.2 Balanceador de carga

9.2.1 Enlaces interesantes balanceador de carga

- maestrodelweb: Balanceo de carga con haproxy
- An Introduction to HAProxy and Load Balancing Concepts
- How To Use HAProxy to Set Up HTTP Load Balancing on an Ubuntu VPS
- Balanceador de carga en openstack

CAPÍTULO 10

Prácticas

10.1 Práctica: Servidor DHCP

Nota: (16 tareas - 25 puntos)(6 tareas obligatorias - 10 puntos)

Nota:

- Muestra al profesor: Tarea 4, Tarea 7, Tarea 13, Tarea 14

10.1.1 Teoría

Nota:

- **Tarea 1 (1 punto):** Lee el documento Teoría: Servidor DHCP y explica el funcionamiento del servidor DHCP resumido en este gráfico.

10.1.2 DHCPv4

Preparación del escenario

Crea un escenario usando Vagrant que defina las siguientes máquinas:

- Servidor: Tiene dos tarjetas de red: una pública y una privada que se conectan a la red local.
- nodo_lan1: Un cliente conectado a la red local.

Servidor dhcp

Instala un servidor dhcp en el ordenador «servidor» que de servicio a los ordenadores de red local, teniendo en cuenta que el tiempo de concesión sea 12 horas y que la red local tiene el direccionamiento 192.168.100.0/24.

Advertencia:

- **Tarea 2 (1 punto)(Obligatorio):** Entrega el fichero Vagrantfile que define el escenario.
- **Tarea 3 (3 puntos)(Obligatorio):** Muestra al profesor el servidor DHCP funcionando. Muestra el fichero de configuración del servidor, la lista de concesiones, la modificación en la configuración que has hecho en el cliente para que tome la configuración de forma automática y muestra la salida del comando *ifconfig*.
- **Tarea 4 (2 puntos):** Muestra al profesor el servidor funcionando como router y NAT, de esta forma los clientes tendrán internet.
- **Tarea 5 (1 punto):** Realizar una captura, desde el servidor usando **tcpdump**, de los cuatro paquetes que corresponden a una concesión: DISCOVER, OFFER, REQUEST, ACK.

Funcionamiento del dhcp

Advertencia: Vamos a comprobar que ocurre con la configuración de los clientes en determinadas circunstancias, para ello vamos a poner un tiempo de concesión muy bajo. Muestra los resultados al profesor.

- **Tarea 6 (2 punto):** Los clientes toman una configuración, y a continuación apagamos el servidor dhcp. ¿qué ocurre con el cliente windows? ¿Y con el cliente linux?
- **Tarea 7 (2 punto)(Obligatorio):** Los clientes toman una configuración, y a continuación cambiamos la configuración del servidor dhcp (por ejemplo el rango). ¿qué ocurriría con un cliente windows? ¿Y con el cliente linux?

Reservas

Crea una reserva para el que el cliente tome siempre la dirección 192.168.100.100.

Advertencia:

- **Tarea 8 (2 puntos)(Obligatorio):** Indica las modificaciones realizadas en los ficheros de configuración y muestra al profesor una comprobación de que el cliente ha tomado esa dirección.

Uso de varios ámbitos

Modifica el escenario Vagrant para añadir una nueva red local y un nuevo nodo:

- Servidor: En el servidor hay que crear una nueva interfaz
- nodo_lan2: Un cliente conectado a la segunda red local.

Configura el servidor dhcp en el ordenador «servidor» para que de servicio a los ordenadores de la nueva red local, teniendo en cuenta que el tiempo de concesión sea 24 horas y que la red local tiene el direccionamiento 192.168.200.0/24.

Advertencia:

- **Tarea 9 (1 punto)**: Entrega el nuevo fichero Vagrantfile que define el escenario.
- **Tarea 10 (1 punto)**: Explica las modificaciones que has hecho en los distintos ficheros de configuración. Entrega las comprobaciones necesarias de que los dos ámbitos están funcionando.
- **Tarea 11 (1 punto)**: Realiza las modificaciones necesarias para que los cliente de la segunda red local tengan acceso a internet. Entrega las comprobaciones necesarias.

10.1.3 DHCPv6

SLAAC

Vamos a usar el primer escenario para configurar en el cliente el programa `radvd` para comprobar como los clientes se autoconfiguran con una dirección ipv6 (SLAAC (Stateless Address Autoconfiguration)). Vamos a trabajar con el prefijo `2001:abcd::/64`.

Advertencia:

- **Tarea 12 (1 punto)(Obligatorio):** Configura de manera adecuada en el servidor el programa *radvd* y comprueba que los clientes (Linux y Windows) se configuran coun ipv6 global.
- **Tarea 13 (1 punto):** Configura *radvd* para entregar también el servidor DNS (RDNSS) y el campo *search* (SNSSL). Comprueba qué esos datos lo configura el cliente Linux. ¿Y el cliente Windows?

DCHPv6

Configura en el servidor isc-dhcp-server una zona para repartir los siguientes elementos:

- Un rango de direcciones: `2001:abcd::1000` hasta `2001:abcd::2000`
- El DNS y el campo *search*.

Advertencia:

- **Tarea 14 (2 puntos)(Obligatorio):** Configura de manera adecuada en el servidor dhcpv6 y comprueba que los clientes (Linux y Windows) se configuran con ipv6 global.
- **Tarea 15 (1 punto):** Configura una reserva para que el cliente linux se configure con la dirección *2001:abcd::a*.

Delegación de prefijo (PD)

En nuestra red tenemos un servidor DHCPv6 puede repartir un prefijo cuando se realiza una petición (es decir cuando se solicita el prefijo), esto nos puede servir para crear un router dentro de nuestra red interna que reparta direcciones ipv6 con un determinado prefijo.

Advertencia:

- **Tarea 16 (3 puntos):** Configura en tu servidor un cliente dibbler-dhcp que es capaz de recoger el prefijo delegado por nuestro servidor *macaco*. Condigura *radvd* para que reparta direcciones con ese prefijo. Comprueba que los clientes (Linux y Windows) se configuran con ipv6 global. Realiza un ping desde el cliente a la dirección *2001:ccba:470::1* que es la de macaco.

10.2 Práctica: Servidor Web Apache 2.4

Nota: (17 tareas - 21 puntos)(8 tareas obligatorias - 11 puntos)

Nota:

- Muestra al profesor: *Tarea 4, Tarea 9 y Tarea 13*

Advertencia:

- **Tarea 1 (1 punto)(Obligatorio):** Crea un escenario Vagrant con una máquina con una red pública. Instala el servidor web Apache2 en la máquina. Modifica la pagína index.html que viene por defecto y accede a ella desde un navegador. Entrega una captura de pantalla accediendo a ella.

10.2.1 Virtual Hosting

Queremos que nuestro servidor web ofrezca dos sitios web, teniendo en cuenta lo siguiente:

1. Cada sitio web tendrá nombres distintos.
2. Cada sitio web compartirán la misma dirección IP y el mismo puerto (80).

Los dos sitios web tendrán las siguientes características:

- El nombre de dominio del primero será `www.iesgn.org`, su directorio base será `/srv/www/iesgn` y contendrá una página llamada `index.html`, donde sólo se verá una bienvenida a la página del Instituto Gonzalo Nazareno.
- En el segundo sitio vamos a crear una página donde se pondrán noticias por parte de los departamento, el nombre de este sitio será `departamentos.iesgn.org`, y su directorio base será `/srv/www/departamentos`. En este sitio sólo tendremos una página inicial `index.html`, dando la bienvenida a la página de los departamentos del instituto.

Advertencia:

- **Tarea 2 (3 punto)(Obligatorio):** Configura la resolución estática en los clientes y muestra el acceso a cada una de las páginas.

10.2.2 Mapeo de URL

Cambia la configuración del sitio web `www.iesgn.org` para que se comporte de la siguiente forma:

Advertencia:

- **Tarea 3 (1 punto)(Obligatorio):** Cuando se entre a la dirección `www.iesgn.org` se redireccionará automaticamente a `www.iesgn.org/principal`, donde se mostrará el mensaje de bienvenida. En el directorio **principal** no se permite ver la lista de los ficheros, no se permite que se siga los enlaces símbolicos y no se permite negociación de contenido. Muestra al profesor el funcionamiento.
- **Tarea 4 (1 punto)(Obligatorio):** Si accedes a la página `www.iesgn.org/principal/documentos` se visualizarán los documentos que hay en /srv/doc. Por lo tanto se permitirá el listado de fichero y el seguimiento de enlaces símbolicos siempre que sean a ficheros o directorios cuyo dueño sea el usuario. Muestra al profesor el funcionamiento.
- **Tarea 5 (1 punto):** En todo el host virtual se debe redefinir los mensajes de error de objeto no encontrado y no permitido. Para el ello se crearan dos ficheros html dentro del directorio error. Entrega las modificaciones necesarias en la configuración y una comprobación del buen funcionamiento.

10.2.3 Autentificación, Autorización, y Control de Acceso

Advertencia:

- **Tarea 6 (1 punto)(Obligatorio):** Añade al escenario Vagrant otra máquina conectada por una red interna al servidor. A la URL `departamentos.iesgn.org/intranet` sólo se debe tener acceso desde el cliente de la red local, y no se pueda acceder desde la anfitriona por la red pública. A la URL `departamentos.iesgn.org/internet`, sin embargo, sólo se debe tener acceso desde la anfitriona por la red pública, y no desde la red local.
- **Tarea 7 (1 punto):** Autentificación básica. Limita el acceso a la URL `departamentos.iesgn.org/secreto`. Comprueba las cabeceras de los mensajes HTTP que se intercambian entre el servidor y el cliente. ¿Cómo se manda la contraseña entre el cliente y el servidor?. Entrega una breve explicación del ejercicio.
- **Tarea 8 (1 punto)(Obligatorio):** Cómo hemos visto la autentificación básica no es segura, modifica la autentificación para que sea del tipo *digest*, y sólo sea accesible a los usuarios pertenecientes al grupo *directivos*. Comprueba las cabeceras de los mensajes HTTP que se intercambian entre el servidor y el cliente. ¿Cómo funciona esta autentificación?
- **Tarea 9 (1 punto):** Vamos a combianar el control de acceso (tarea 6) y la autentificación (tareas 7 y 8), y vamos a configurar el virtual host para que se comporte de la siguiente manera: el acceso a la URL `departamentos.iesgn.org/secreto` se hace forma directa desde la intranet, desde la red pública te pide la autentificación. Muestra el resultado al profesor.

10.2.4 Configuración con .htaccess

Date de alta en un **proveedor de hosting**. ¿Si necesitamos configurar el servidor web que han configurado los administradores del proveedor?, ¿qué podemos hacer? Explica la directiva `AllowOverride` de apache2. Utilizando archivos .htaccess realiza las siguientes configuraciones:

Advertencia:

- **Tarea 10 (1 punto)(Obligatorio):** Habilita el listado de ficheros en la URL `http://host.dominio/nas`.

- **Tarea 11 (1 punto):** Crea una redirección permanente: cuando entremos en `ttp://host.dominio/google` salte a `ww.google.es`.
- **Tarea 12 (1 punto):** Pedir autentificación para entrar en la URL `http://host.dominio/prohibido`. (No la hagas si has elegido como porveedor CDMON, en la plataforma de prueba no funciona.)

10.2.5 Módulos

Advertencia:

- **Tarea 13 (2 puntos)(Obligatorio):** Módulo *userdir*: Activa y configura el módulo *userdir*, que permite que cada usuario del sistema tenga la posibilidad de tener un directorio (por defecto se llama `public_html`) donde alojar su página web. Publica una página de un usuario, y accede a la misma. Esta tarea la tienes que hacer en tu servidor.
- **Tarea 14 (2 puntos):** En tu servidor crea una carpeta `php` donde vamos a tener un fichero `index.php` con el siguiente contenido:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.
↪org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Conversor de Monedas</title>
</head>

<body>
<form action="index.php" method="get">
    <input type="text" size="30" name="monto" /><br/>
    <select name="pais">
        <option name="Dolar">Dolar</option>
        <option name="Libra">Libra</option>
        <option name="Yen">Yen</option>
    </select>
    <input type="submit" value="convertir" />
   </form>
<?php
    // averiguamos si se ingresó un motno
    if (isset($_GET['monto'])) {
      define ("cantidad", $_GET['monto']);
    } else {
      define ("cantidad", 0);
    }
    if($_GET){
    // definimos los paises
    $tasacambios = array ("Libra"=>0.86,"Dolar"=>1.34,"Yen"=>103.56);
    // imprimimos el monto ingresado
    echo "<b>".cantidad." euros</b><br/> ".$_GET["pais"]." = ".cantidad*
↪$tasacambios[$_GET["pais"]]." <br><br>";
    // por cada pais imprimimos el cambio
    }
   ?>

</body>
</html>
```

Prueba la página utilizando parámetros en la URL (parámetros GET), por ejemplo: `http://nombre_página/php/index.php?monto=100&pais=Libra`

Configura mediante un fichero `.htaccess`, la posibilidad de acceder a la URL **http://nombre_página/php/moneda/cantidad**, donde moneda indica el nombre de la moneda a la que queremos convertir (Dolar,Libra,Yen) y cantidad indica los euros que queremos convertir.

10.2.6 IPv6

Advertencia:

- **Tarea 15 (1 punto)**: Comprueba que el servidor web con la configuración por defecto está escuchando por el puerto 80 en ipv6.
- **Tarea 16 (1 punto)**: Configura la máquina para que tenga una ipv6 global. Activa el virtualhost por defecto y accede a la página principal utilizando la ipv6 global que tiene asignada.
- **Tarea 17 (1 punto)**: Configura la resolución estática para acceder a los virtualhost utilizando ipv6.

10.3 Práctica: Servidor Web Nginx

Nota: (12 tareas - 13 puntos)(6 tareas obligatorias - 8 puntos)

Nota:

- Muestra al profesor: *Tarea 4, Tarea 9*

Advertencia:

- **Tarea 1 (1 punto)(Obligatorio):** Crea un escenario Vagrant con una máquina con una red pública. Instala el servidor web nginx en la máquina. Modifica la paǵina index.html que viene por defecto y accede a ella desde un navegador. Entrega una captura de pantalla accediendo a ella.

10.3.1 Virtual Hosting

Queremos que nuestro servidor web ofrezca dos sitios web, teniendo en cuenta lo siguiente:

1. Cada sitio web tendrá nombres distintos.
2. Cada sitio web compartirán la misma dirección IP y el mismo puerto (80).

Los dos sitios web tendrán las siguientes características:

- El nombre de dominio del primero será `www.iesgn.org`, su directorio base será `/srv/www/iesgn` y contendrá una página llamada `index.html`, donde sólo se verá una bienvenida a la página del Instituto Gonzalo Nazareno.

- En el segundo sitio vamos a crear una página donde se pondrán noticias por parte de los departamento, el nombre de este sitio será `departamentos.iesgn.org`, y su directorio base será `/srv/www/departamentos`. En este sitio sólo tendremos una página inicial `index.html`, dando la bienvenida a la página de los departamentos del instituto.

Advertencia:

- **Tarea 2 (3 punto)(Obligatorio):** Configura la resolución estática en los clientes y muestra el acceso a cada una de las páginas.

10.3.2 Mapeo de URL

Cambia la configuración del sitio web `www.iesgn.org` para que se comporte de la siguiente forma:

Advertencia:

- **Tarea 3 (1 punto)(Obligatorio):** Cuando se entre a la dirección `www.iesgn.org` se redireccionará automaticamente a `www.iesgn.org/principal`, donde se mostrará el mensaje de bienvenida. En el directorio **principal** no se permite ver la lista de los ficheros, no se permite que se siga los enlaces símbolicos y no se permite negociación de contenido. Muestra al profesor el funcionamiento.
- **Tarea 4 (1 punto)(Obligatorio):** Si accedes a la página `www.iesgn.org/principal/documentos` se visualizarán los documentos que hay en /srv/doc. Por lo tanto se permitirá el listado de fichero y el seguimiento de enlaces símbolicos siempre que sean a ficheros o directorios cuyo dueño sea el usuario. Muestra al profesor el funcionamiento.
- **Tarea 5 (1 punto):** En todo el host virtual se debe redefinir los mensajes de error de objeto no encontrado y no permitido. Para el ello se crearan dos ficheros html dentro del directorio error. Entrega las modificaciones necesarias en la configuración y una comprobación del buen funcionamiento.

10.3.3 Autentificación, Autorización, y Control de Acceso

Advertencia:

- **Tarea 6 (1 punto)(Obligatorio):** Añade al escenario Vagrant otra máquina conectada por una red interna al servidor. A la URL `departamentos.iesgn.org/intranet` sólo se debe tener acceso desde el cliente de la red local, y no se pueda acceder desde la anfitriona por la red pública. A la URL `departamentos.iesgn.org/internet`, sin embargo, sólo se debe tener acceso desde la anfitriona por la red pública, y no desde la red local.
- **Tarea 7 (1 punto):** Autentificación básica. Limita el acceso a la URL `departamentos.iesgn.org/secreto`. Comprueba las cabeceras de los mensajes HTTP que se intercambian entre el servidor y el cliente. ¿Cómo se manda la contraseña entre el cliente y el servidor?. Entrega una breve explicación del ejercicio.
- **Tarea 8 (1 punto)(Obligatorio):** Cómo hemos visto la autentificación básica no es segura, modifica la autentificación para que sea del tipo *digest*, y sólo sea accesible a los usuarios pertenecientes al grupo *directivos*. Comprueba las cabeceras de los mensajes HTTP que se intercambian entre el servidor y el cliente. ¿Cómo funciona esta autentificación?
- **Tarea 9 (1 punto):** Vamos a combianar el control de acceso (tarea 6) y la autentificación (tareas 7 y 8), y vamos a configurar el virtual host para que se comporte de la siguiente manera: el acceso a la URL

`departamentos.iesgn.org/secreto` se hace forma directa desde la intranet, desde la red pública te pide la autentificación. Muestra el resultado al profesor.

10.3.4 IPv6

Advertencia:

- **Tarea 10 (1 punto)**: Comprueba que el servidor web con la configuración por defecto está escuchando por el puerto 80 en ipv6.
- **Tarea 11 (1 punto)**: Configura la máquina para que tenga una ipv6 global. Activa el virtualhost por defecto y accede a la página principal utilizando la ipv6 global que tiene asignada.
- **Tarea 12 (1 punto)**: Configura la resolución estática para acceder a los virtualhost utilizando ipv6.

10.4 Práctica: Servidor DNS

Nota: (8 tareas - 25 puntos)(3 tareas obligatorias - 10 puntos)

Nota:

- Muestra al profesor: *Tarea 2, Tarea 6 y Tarea 7*

10.4.1 Escenario

1. En nuestra red local tenemos un **servidor Web** que sirve dos páginas web: *www.iesgn.org*, *departamentos.iesgn.org*
2. Vamos a instalar en nuestra red local un servidor DNS (lo puedes instalar en el mismo equipo que tiene el servidor web)
3. Voy a suponer en este documento que el nombre del servidor DNS va a ser `pandora.iesgn.org`. Si quieres puedes utilizar otro nombre.

10.4.2 Servidor DNSmasq

Instala el servidor dns **dnsmasq** en `pandora.iesgn.org` y configúralo para que los clientes puedan conocer los nombres necesarios.

Advertencia:

- **Tarea 1 (2 punto)(Obligatorio):** Modifica los clientes para que utilicen el nuevo servidor dns. Realiza una consulta a www.iesgn.org, y a www.josedomingo.org. Realiza una prueba de funcionamiento para comprobar que el servidor dnsmasq funciona como cache dns. Muestra el fichero `hosts` del cliente para demostrar

que no estás utilizando resolución estática. ¿Se puede realizar resolución inversa?. Documenta la tarea en redmine.

10.4.3 Servidor bind9

Desinstala el servidor **dnsmasq** del ejercicio anterior e instala un servidor dns **bind9**. Las características del servidor DNS que queremos instalar son las siguientes:

- El servidor DNS se llama `pandora.iesgn.org` y por supuesto, va a ser el servidor con autoridad para la zona `iesgn.org`.
- Vamos a suponer que tenemos un servidor para recibir los correos que se llame `correo.iesgn.org` y que está en la dirección x.x.x.200 (esto es ficticio).
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.iesgn.org` y que está en x.x.x.201 (esto es ficticio)
- Además queremos nombrar a los clientes.
- También hay que nombrar a los virtual hosts de apache: `www.iesgn.org` y `departementos.iesgn.org`
- Se tienen que resolver las direcciones ipv6 de las distintas máuinas (inventante las ficticias).
- Se tienen que configurar la zona de resolución inversa.

Advertencia:

- **Tarea 2 (4 puntos)(Obligatorio):** Realiza la instalación y configuración del servidor bind9 con las características anteriomente señaladas. Entrega las zonas que has definido.
- **Tarea 3 (4 puntos)(Obligatorio): Realiza las consultas dig/nslookup desde los clientes preguntando por los siguientes:**
  - Dirección de `pandora.iesgn.org`, `www.iesgn.org`, `ftp.iesgn.org`
  - El servidor DNS con autoridad sobre la zona del dominio `iesgn.org`
  - El servidor de correo configurado para `iesgn.org`
  - La dirección IP de `www.josedomingo.org`
  - Un resolución inversa
  - La dirección ipv6 de `pandora.iesgn.org`

10.4.4 Servidor DNS esclavo

El servidor DNS actual funciona como **DNS maestro**. Vamos a instalar un nuevo servidor DNS que va a estar configurado como **DNS esclavo** del anterior, donde se van a ir copiando periódicamente las zonas del DNS maestro. Suponemos que el nombre del servidor DNS esclavo se va llamar `afrodita.iesgn.org`.

Advertencia:

- **Tarea 4 (3 puntos): Realiza la instalación del servidor DNS esclavo. Documenta los siguientes apartados:**
  - Entrega la configuración de las zonas del maestro y del esclavo.

- Comprueba si las zonas definidas en el maestro tienen algún error con el comando adecuado.
- Comprueba si la configuración de `named.conf` tiene algún error con el comando adecuado.
- Reinicia los servidores y comprueba en los logs si hay algún error. **No olvides incrementar el número de serie en el registro SOA si has modificado la zona en el maestro**.
- Muestra la salida del log donde se demuestra que se ha realizado la transferencia de zona.

- **Tarea 5 (3 puntos): Documenta los siguientes apartados:**
  - Configura un cliente para que utilice los dos servidores como servidores DNS.
  - Realiza una consulta con `dig` tanto al maestro como al esclavo para comprobar que las respuestas son autorizadas. ¿En qué te tienes que fijar?
  - Solicita una copia completa de la zona desde el cliente ¿qué tiene que ocurrir?. Solicita una copia completa desde el esclavo ¿qué tiene que ocurrir?
- **Tarea 6 (2 puntos): Muestra al profesor el funcionamiento del DNS esclavo:**
  - Realiza una consulta desde el cliente y comprueba que servidor está respondiendo.
  - Posteriormente apaga el servidor maestro y vuelve a realizar una consulta desde el cliente ¿quién responde?

10.4.5 Delegación de dominios

Tenemos un servidor DNS que gestiona la zona correspondiente al nombre de dominio `iesgn.org`, en esta ocasión queremos delegar el subdominio `informatica.iesgn.org` para que lo gestione otro servidor DNS. Por lo tanto tenemos un escenario con dos servidores DNS:

- `pandora.iesgn.org`, es servidor DNS autorizado para la zona `iesgn.org`.
- `ns.informatica.iesgn.org`, es el servidor DNS para la zona `informatica.iesgn.org` y, está instalado en otra máquina.

Los nombres que vamos a tener en ese subdominio son los siguientes:

- `www.informatica.iesgn.org` corresponde a un sitio web que está alojado en el servidor web del departamento de informática.
- Vamos a suponer que tenemos un servidor ftp que se llame `ftp.informatica.iesgn.org` y que está en la misma máquina.
- Vamos a suponer que tenemos un servidor para recibir los correos que se llame `correo.informatica.iesgn.org`.

Advertencia:

- **Tarea 7 (4 puntos):** Realiza la instalación y configuración del nuevo servidor dns con las características anteriormente señaladas. Muestra el resultado al profesor.
- **Tarea 8 (3 puntos):** Realiza las consultas dig/neslookup desde los clientes preguntando por los siguientes:
  - Dirección de `www.informatica.iesgn.org`, `ftp.informatica.iesgn.org`
  - El servidor DNS que tiene configurado la zona del dominio `informatica.iesgn.org`. ¿Es el mismo que el servidor DNS con autoridad para la zona `iesgn.org`?
  - El servidor de correo configurado para `informatica.iesgn.org`

10.5 Práctica: Configuración de servidores GNU/Linux

Nota: (12 tareas - 20 puntos)(6 tareas obligatorias - 7 puntos)

Advertencia: Objetivo

El objetivo de esta práctica es montar una infraestrucuta de servicios que se mantenga en el tiempo y que nos sirva para montar servicios y aplicaciones en los distintos módulos durante el curso. Esta práctica la tenéis que realizar en la infraestructura de máquinas que hemos creado en el cloud para todas los módulos. En cualquier momento del curso los servicios que instalemos en esta práctica deben estar funcionando de manera adecuada.

- Servidor1: mickie (Debian)
- Servidor2: minnie (Ubuntu)
- Servidor3: donald (CentOs)

Advertencia: Ejemplo de nombres, suponiendo que mi nombre de dominio va a ser `josedom.gonzalonazareno.org`:

Los nombres de los equipos van a ser:

- `minnie.josedom.gonzalonazareno.org`
- `mickie.josedom.gonzalonazareno.org`
- `donald.josedom.gonzalonazareno.org`
- El servidor DNS va a estar instalado en `mickie.josedom.gonzalonazareno.org`
- El servidor web va a estar instalado en `donald.josedom.gonzalonazareno.org`, y vamos a tener dos páginas webs:
  - `www.josedom.gonzalonazareno.org`
  - `informatica.josedom.gonzalonazareno.org`
- El servidor de base de datos va a estar instalado en `minnie.josedom.gonzalonazareno.org`

10.5.1 Servidor DNS

Vamos a instalar un sevidor dns que nos permita gestionar la resolución directa e inversa de nuestros nombres. Cada alumno va a poseer un servidor dns con autoridad sobre un subdominio de nuestro dominio principal *gonzalonazareno.org*, que se llamará `tu_nombre.gonzalonazareno.org`.

Nota: Indica al profesor el nombre de tu dominio para que pueda realizar la delegación en el servidor DNS principal *papion*.

Instalación del servidor DNS

```
El servidor DNS se va a instalar en el servidor1 (mickie). Y en un primer momento se␣
↪configurará de la siguiente manera:

    * El servidor DNS se llama ``mickie.tu_nombre.gonzalonazareno.org`` y va a ser el␣
↪servidor con autoridad para la zona ``tu_nombre.gonzalonazareno.org``.
    * El servidor debe resolver el nombre de los tres servidores.
    * Se debe configurar las zonas de resolución inversa.
```

Nota:

- Debes determinar si la resolución directa se hace con dirección ip fijas o flotantes del cloud depediendo del servicio que se este prestando.
- Debes considerar la posibilidad de hacer dos zonas de resolución inversa para resolver ip fijas o flotantes del cloud.
- Debes modificar la configuración del servidor DHCP del cloud para que mande a los servidores el nuevo nombre de dominio.

Advertencia:

- **Tarea 1 (1 puntos):** Comprueba que los servidores tienen configurados el nuevo nombre de dominio de forma adecuada después de volver a renovar la concesión del servidor DHCP. Documenta el contenido del fichero en el que se puede comprobar este punto (ejecuta el comando `hostname -f` y muestra el fichero `/etc/resolv.conf`).
- **Tarea 2 (2 puntos)(Obligatorio):** Entrega el resultado de las siguientes consultas :
  - El servidor DNS con autoridad sobre la zona del dominio `tu_nombre.gonzalonazareno.org`
  - La dirección IP de los tres servidores
  - Un resolución inversa de IP fija, y otra resolución inversa de IP flotante.

```
Nos gustaría poder dar de alta nuevos nombres en el servidor DNS. Para ello vas a␣
↪crear un scipt en python que nos permita añadir o borrar registros en las zonas de␣
↪nuestro servidor.

El script se debe llamar ``gestionDNS.py`` y recibe cutro parámetros:

    * ``-a`` o ``-b``: Si recibe ``-a`` añadirá un nuevo nombre, si recibe ``-b``␣
↪borrará el nombre que ha recibido.
    * ``-dir`` o ``-alias``, si recibe ``-dir`` va a crear un registro tipo A, si␣
↪recibe ``-alias`` va a crear un registro CNAME
    * El nombre de la máquina para añadir o borrar
    * El nombre del alias o la dirección ip: Si has usuado la opción ``-dir``␣
↪recibirá una ip y si has usuado ``-alias`` recibirá el nombre de la máquina a la␣
↪que le vamos a hacer el alias. Si has utilizado -b no teendrá este parámetro.

Ejemplos

    ``gestionDNS.py -a -dir smtp 192.168.4.1``

    Creará el registro -> smtp    A    192.168.4.1
```

(continues on next page)

(proviene de la página anterior)

```
    ``gestionDNS.py -a -alias correo smtp``

    Creará el registro -> correo      CNAME    smtp

    ``gestionDNS.py -b correo``

    Borrará el último registro

Todos los registros creados o borrados pertenecen a las zonas ``tu_nombre.
↪gonzalonazareno.org``. Se debe modificar la zona inversa en los casos necesarios.␣
↪El script debe reinciar el servidor bind9.
```

Advertencia:

- **Tarea 3 (3 puntos):** Entrega el repositorio github donde has desarrollado el script y realiza un ejemplo al profesor.

10.5.2 Servidor Web

En nuestro servidor3 (donald) vamos a instalar un servidor Web apache2 con las siguientes características.

Advertencia:

- **Tarea 4 (1 punto)(Obligatorio):** Nuestro servidor va a tener dos hosts virtuales: `www.tu_nombre.gonzalonazareno.org` y `informatica.tu_nombre.gonzalonazareno.org`. Explica los pasos fundamentales para realizar los dos virtual hosts.
- **Tarea 5 (1 punto):** Comenta los cambios en el servidor DNS para de dar de alta los dos nuevos nombres.
- **Tarea 6 (1 punto)(Obligatorio):** La página `www.tu_nombre.gonzalonazareno.org` va a ser la página principal, busca una plantilla html, modifícala un poco y despliegala en el primer virtual host. Muestrasela al profesor.
- **Tarea 7 (1 punto)(Obligatorio):** Por seguridad, en la página `www.tu_nombre.gonzalonazareno.org`, no se permite que se sigan enlaces simbólicos, no se permite negociación de contenidos, no se permite visualizar la lista de ficheros y no se permite usar ficheros .htaccess. Entrega la modificaciones en la configuración necesarias.
- **Tarea 8 (1 punto)(Obligatorio):** La página `informatica.tu_nombre.gonzalonazareno.org` es una página relacionada con el mundo de la informática, busca una plantilla html, modifícala un poco y despliegala en el primer virtual host. La página se guardará en un directorio llamado plataforma. Por lo tanto si accedemos a `informatica.example.com` se debererá redirigir automáticamente a `informatica.example.com/plataforma`. Muestra el resultado al profesor.
- **Tarea 9 (3 puntos):** Para llevar una estadistica de visitas y accesos instala la aplicación awstats en el servidor. Configura el cron para que la estadistíca se vaya actualizando. Debes realizar dos estadísticas, una para cada host virtual.
- **Tarea 10 (3 puntos):** En el directorio `/srv/isos` tenemos una colección de imágenes isos, queremos acceder a ella en la dirección `informatica.tu_nombre.gonzalonazareno.org/isos`. Esta dirección debe ser sólo accesible desde la intranet, si accedemos desde fuera tenemos que autentificarnos (digest) con un usuario.

10.5.3 Servidor de Base de Datos

En nuestro servidor2 (minnie) vamos a instalar un servidor de base de datos mysql.

Advertencia:

- **Tarea 11 (1 punto)(Obligatorio):** Configura el servidor para que sea accesible por los equipos de la red local. Muestra al profesor una conexión a la base de datos desde el servidor3 (donald).
- **Tarea 12 (2 puntos):** Instala en el servidor3 (donald) la aplicación phpmyadmin que nos permite gestionar las bases de datos de nuestro servidor. Esta aplicación sólo será accesible desde la URL `www.tu_nombre.gonzalonazareno.org/basededatos`. Muestra el acceso al profesor.

10.6 Práctica: Gestionar un hosting por ftp

Nota: (5 tareas - 15 puntos)(2 tareas obligatorias - 5 puntos)

Nota:

- Cuando termines las tareas tienes que realizar una prueba de funcionamiento al profesor.

Tenemos que desarrollar la página del instituto `www.iesgn.org`, y queremos que sea gestionada por medio de un ftp. Tendremos las siguientes funcionalidades:

- Queremos ofrecer una colección de documentos, y lo vamos a hacer mediante http y ftp anónimo, de esta forma se accederá al mismo directorio si accedo a las siguientes URL:
  - `http://www.iesgn.org/documentos`
  - `ftp://ftp.iesgn.org`

Advertencia:

- **Tarea 1 (2 puntos)(Obligatorio):** Configura apache2 y el servidor ftp de forma anónimo para obtener el resultado pedido. Entrega la configuración de ambos servidores y muestra al profesor el funcionamiento.

- Cada departamento tendrá su página web en la URL `www.iesgn.org/nombredeldepartamento`, veamos un ejemplo:
  - El departamento de matemáticas tendrá su página en `www.iesgn.org/matematicas`
  - Se creará un usuario `user_matematicas`, que tendrá una contraseña, para que accediendo a `ftp.iesgn.org`, pueda gestionar los ficheros de su página web.

  Determina los cambios que tienes que ir realizando para ir creando el espacio web para cada uno de los departamentos.

Advertencia:

- **Tarea 2 (3 puntos)(Obligatorio):** Entrega una pequeña guía que explique las acciones que hay que realizar para crear la página web de un determinado departamento. Muestra al profesor el funcionamiento para la asignatura de «informática».

- Escribe un pequeño script `crear_pagina_web`, que recibe como parámetro el nombre del departamento y realiza los pasos para crear la página web de departamento y que un usuario accediendo por ftp pueda gestionarla.

Advertencia:

- **Tarea 3 (3 puntos):** Entrega la url del repositorio github, y haz una prueba al profesor.

- Instala la aplicación web net2ftp en el servidor por si tenemos problemas de acceso por el puerto 20/21.

Advertencia:

- **Tarea 4 (3 puntos):** Muestra al profesor al acceso a la aplicación web net2ftp.

- El uso de usuarios reales del sistema para el acceso FTP puede tener varias desventajas (gestión, seguridad,...). Modifica la configuración del sistema para que se usen usuarios virtuales para el acceso por FTP, cuya información este guardada en una tabla mysql o en un directorio ldap.

Advertencia: Tarea 5 (4 puntos): Entrega los pasos más relevantes para realizar esta tarea. Y muestra al profesor su funcionamiento.

10.7 Práctica: Implantación de un servidor de hosting

Nota:

- La realización de los apartados «Especificaciones técnicas mínimas» y «Creación de scripts» valen 5 puntos.
- Por cada mejora se sumará un punto. (Hasta 5 puntos)
- Se puede proponer nuevas propuestas.
- Se tendrán que entrega una memoria donde se explique el trabajo que se ha realizado (5 puntos).
- Se entregará una presentación, que usará el grupo para la presentación en público que se realizasrá en clase (5 puntos)

El objetivo de la práctica es montar un servidor que ofrezca un servicio de de hospedaje de páginas web con las siguientes características:

- Podemos dar de alta a un usuario y al nombre de dominio (por ejemplo `nombrededominio.com`) por el que va estar referido su espacio.
- Se podrán hospedar páginas estáticas (html) y páginas web dinámicas construidas con PHP.
- Automáticamente se creará una página principal, que al acceder a la pagina web (`www.nombrededominio.com`) de la bienvenida e informe que dicha página está en construcción.

- Para gestionar los ficheros hospedados en nuestro espacio utilizaremos un servidor FTP `ftp.nombrededominio.com`.
- Para gestionar las tablas de mysql accederemos al programa phpmyadmin en la dirección `mysql.nombrededominio.com`.

Advertencia: Quizás os pueda ayudar a empezar la práctica contestar las siguientes preguntas:

1. ¿Qué servidores necesitas instalar en la máquina donde vamos a implantar el hosting? Cuando damos de alta una nueva cuenta en nuestro hosting hay que indicar un usuario y un nombre de dominio. ¿Qué acciones hay que hacer en el servidor con el usuario? ¿Qué acciones hay que hacer con el nombre de dominio?
2. ¿Cómo puedes comprobar que existe ya un usuario con el mismo nombre?¿y qué ya está dado de alta un determinado nombre de dominio?
3. ¿Qué debes tener en cuenta, a la hora de crear el directorio home del usuario, para que accediendo por ftp, el usuario pueda gestionar su página web?
4. ¿Cuantos nombres habrá que dar de alta en la zona de resolución directa de nombrededominio.com?

10.7.1 Especificaciones técnicas mínimas

- El sistema utilizará usuarios virtuales cuya información estará guardad en una base de datos mysql o en un servidor ldap.
- El administrador debe decidir la estructura para guardar los directorios personales de los usuarios. Cuando se da de alta un nuevo usuario con un nombre de dominio, habrá que tener en cuenta las siguientes consideraciones:

1. Si el usuario o el nombre del dominio existen, no se continúa.
2. Se creará el directorio personal del usuario, este directorio será el `DocumentRoot` del servidor web. En este directorio se tendrá que crear una página web de bienvenida.
3. Se creará un nuevo virtual hosting (`www.nombrededomino.com`) con el `DocumentRoot` apuntando al directorio personal que anteriormente hemos instalado.
4. Se creará un nuevo usuario virtual para el acceso por FTP. El administrador decidirá la política para generar la contraseña. Dicha contraseña generada tendrá que visualizarse por pantalla. La contraseña será guardada en la base de datos encriptada.
5. Se creará un nuevo usuario en el gestor de base de datos mysql, se debe llamar `mynombredeusuario`, la contraseña que se genere para mysql debe ser distinta a la generada para la gestión del FTP y también se debe mostrar.
6. Se creará una nueva zona `nombrededominio.com` en el servidor DNS bind9 con las zonas de resolución directa e inversa que permitan conocer los distintos nombres (www,ftp, mysql, . . .)

10.7.2 Creación de scripts

Crea los siguientes scripts:

- Un script bash/python (`alta`) que reciba el nombre del usuario y el nombre de dominio relacionado con el usuario, y realice los pasos mostrados anteriormente.
- Un script bash/python (`baja`) que reciba un nombre de dominio e elimine la cuenta del usuario relacionado a dicho nombre de dominio. Borrará el vitual host de apache2, la zona del servidor DNS, el usuario de la base de datos y las bases de datos creados, el usuario virtual para el acceso a la base de datos y el directorio personal del usuario.

- Un script bash/python (`change_password`) que nos permite cambiar las contraseñas de un determinado usuario. Por lo tanto recibe el nombre de un usuario, una opción (`-sql`, si queremos cambiar la contraseña de mysql, o `-ftp`, si queremos cambar la contraseña del ftp) y un nueva contraseña y haga la modificación de la contraseña indicada si el usuario existe.

10.7.3 Posibles mejoras

Configuración de estadísticas Webs

Configura el sistema para que todos los usuarios puedan acceder a las estadísticas de su alojamiento web usando el programa awstats. Tendremos que tener en cuanta que el acceso a esta información no será público, para acceder a ella el usuario se tendrá que autentificar con el nombre de usuario y la contraseña que se han generado para la gestión ftp. Crea un sistema de estadística con awstats que sea accesible desde la URL: `stats.nombrededominio.com`.

Utilización de cuotas

Investiga la forma de limitar el espacio que los usuarios tienen a su disposición, por ejemplo que cada usuario tenga un espacio limitado de 100 Mb.

Usuarios virtuales con LDAP

Modifica toda la configuración para que los suarios virtuales que estamos usando se guarden en un servidor LDAP.

Aplicación web para la gestión del hosting

Crea una palicación web (con cualquier tecnología, por ejemplo bottle o django) que permita gestionar el hosting: dar de alta nuevas cuentas, borrarlas, modificar contraseñas,...

Creación de subdominios

Queremos añadir a nuestro hosting para que podamos dar de alta a nuevos subdominios. Por ejemplo podemos crear el subdominio `web.nombrededominio.com` que será otra página web del usuario. Crea un script que nos permita gestionar subdominios en nuestro hosting.

10.8 Práctica: Configuración de un servidor Windows Server 2012

Nota: (8 tareas - 20 puntos)

10.8.1 Esquema de red

Debes configurar en un entorno virtual en KVM donde tengas 3 máquinas:

- Un servidor Windows Server 2012 con dos interfaces de red, uno conectado a internet, y otro a una red intena donde se encuentran los clientes.
- Dos clientes: uno Linux y otro Windows.

Advertencia:

- **Tarea 1 (2 puntos):** Configuración de la red virtual en KVM. Indica el nombre que le has puesto al ordenador servidor. Y por último, indica la configuración de red del servidor y de algún cliente.
- **Tarea 2 (2 puntos):** Explica la configuración del servidor para que funcione como router y nat.

10.8.2 Servidor DHCP

Los ordenadores clientes de nuestra LAN obtienen su configuración de red ofrecidas por dicho servidor, que tiene las siguientes características:

```
* Rango: 192.168.1.3-192.168.1.254
* Máscara de red: 255.255.255.0
* Puerta de enlace: La ip del router
* DNS: Según el que te convenga para hacer las pruebas
```

Crea una reserva para el que uno de los clientes tome siempre una dirección fija.

Advertencia:

- **Tarea 3 (3 puntos):** Muestra al profesor el servidor DHCP funcionando. Muestra el fichero de configuración del servidor, la lista de concesiones, la modificación en la configuración que has hecho en el cliente para que tome la configuración de forma automática y muestra la salida del comando ifconfig.
- **Tarea 4 (2 puntos):** Indica las modificaciones realizadas en el servidor y muestra al profesor una comprobación de que el cliente ha tomado esa dirección.

10.8.3 Servidor DNS

Lo primero que tienes que hacer es determinar un nombre de dominio que va a ser utilizado en nuestro sistema. (En esta documentación voy a utilizar el nombre `example.com`). El servidor DNS ofrece el servicio de resolución de nombres para los ordenadores de nuestra red local. Debes tener en cunta los siguientes puntos:

- Cuando tengas funcionando el servidor DNS, tendrás que modificar el servidor DHCP para que los clientes usen el nuevo servidor DNS.
- Piensa el nombre que tiene el servidor. El servidor DNS debe poder resolver los siguientes nombres: `nombredelservidor.example.com`, `www.example.com`, `informatica.example.com`. El primero es el nombre del servidor, los dos siguientes son dos páginas webs que el servidor va a servir.
- Debes implementar la zona inversa del servidor.

Advertencia:

- **Tarea 5 (3 puntos):** Realiza la instalación y configuración del servidor DNS con las características anteriomente señaladas. Indica el cambio que hay que hacer en el servidor dhcp para que el sistema funcione de manera adecuada. Entrega una captura de pantalla donde se vea las zonas que has definido. Muestra el resultado al profesor.
- **Tarea 6 (3 puntos):** Realiza las consultas dig/nslookup desde los clientes preguntando por los siguientes:
- Dirección de `www.example.com`
- El servidor DNS con autoridad sobre la zona del dominio
- La dirección IP de `www.josedomingo.org`
- Una resolución inversa

10.8.4 Servidor WEB

El servidor tiene instalado un servidor Web IIS, que sirve dos virtual host con páginas web estáticas.

- `www.example.com`: Página principal del centro.
- `informatica.example.com`: Página del departamento de informática
- La página `www.example.com`, posee un directorio /privado, que para acceder a el es necesario autentificarse.

Advertencia:

- **Tarea 7 (3 puntos):** Configura el servidor web para servir las dos páginas. Busca plantillas html+css como contenido para las dos páginas, modificandolas un poco para que parezcan más reales. Muestra las páginas al profesor.
- **Tarea 8 (2 puntos):** Autentificación de la carpeta /privado: Sólo se pueden autentificar los usuarios del sistemas que pertenezcan al grupo profesores. Muestra el funcionamiento al profesor.

10.9 Práctica: Ejecución de script PHP. Rendimiento.

Nota: (9 tareas - 20 puntos)(8 tareas obligatorias - 15 puntos)

Vamos a comparar el rendimiento de distintas configuraciones de servidores web sirviendo páginas dinámicas programadas con PHP, en concreto vamos a servir un CMS Wordpress.

Las configuraciones que vamos a realizar son las siguientes:

- Módulo php5-apache2
- PHP-FPM + apache2
- PHP-FPM + nginx

Nota:

- **Tarea 1 (1 punto)(Obligatorio)**: Documenta la instalación del módulo php de apache2. Muestra wordpress funcionando con el módulo php de apache2. Realiza una comprobación de que, efectivamente, se está usando el módulo php.
- **Tarea 2 (1 punto)(Obligatorio)**: Documenta la instalación y configuración de FPM-PHP y apache2 (escuchando en un socket UNIX) con el módulo de multiprocesamiento event. Muestra wordpress funcionando con FPM-PHP. Realiza una comprobación de que, efectivamente, se está usando FPM-PHP.
- **Tarea 3 (1 punto)(Obligatorio)**: Cambia la configuración anterior para que PHP-FPM escuche en un socket TCP.
- **Tarea 4 (1 punto)(Obligatorio)**: Documenta la instalación y configuración de PHP-FPM y nginx (escuchando en un socket UNIX) con el módulo de multiprocesamiento event. Muestra wordpress funcionando con PHP-FPM. Realiza una comprobación de que, efectivamente, se está usando PHP-FPM.
- **Tarea 5 (1 punto)(Obligatorio)**: Cambia la configuración anterior para que PHP-FPM escuche en un socket TCP.

10.9.1 Estudio de rendimiento

Ahora utilizando el script benchmark.py, realiza las pruebas de rendiemento para cada una de las configuraciones anteriores:

- Módulo php5-apache2
- FPM-PHP + apache2 (escuchando en un socket UNIX o en un socket TCP)
- FPM-PHP + nginx (escuchando en un socket UNIX o en un socket TCP)

Nota:

- **Tarea 6 (5 puntos)(Obligatorio)**: Entrega la configuración del script de pruebas para cada una de las configuraciones. Entrega los datos obtenidos y la gráfica que has generado.

10.9.2 Aumento de rendimiento

Nota:

- **Tarea 7 (2 puntos)(Obligatorio)**: Añade a la configuración ** ganadora del punto anterior** memcached. Documenta la instalación y configuración memcached. Recuerda que para que Wordpress utilice memcached le tenemos que instalar un plugin. Muestra las estadísticas de memcached después de acceder varias veces a wordpress para comprobar que esa funcionando.
- **Tarea 8 (3 puntos)(Obligatorio)**: Configura un proxy inverso - caché Varnish escuchando en el puerto 80 y que se comunica con el servidor web por el puerto 8080. Entrega y muestra una comprobación de que varnish está funcionando con la nueva configuración.

10.9.3 Estudio de rendimiento

A continuacion hacemos el estudio de rendimiento para las siguientes configuraciones:

- Configuración ganadora
- Configuración ganadora + memcached
- Configuración ganadora + varnish

Nota:

- **Tarea 9 (5 puntos)**: Entrega la configuración del script de pruebas para cada una de las configuraciones. Entrega los datos obtenidos y la gráfica que has generado.

10.10 Práctica: Ejecución de script Python. Rendimiento.

Nota: (6 tareas - 15 puntos)(3 tareas obligatorias - 6 puntos)

Vamos a comparar el rendimiento de distintas configuraciones de servidores web sirviendo páginas dinámicas programadas con Python, en concreto vamos a servir un CMS Mezzanine (Instala algunas páginas de demostración durente la instalación: Would you like to install some initial demo pages?).

Las configuraciones que vamos a realizar son las siguientes:

- apache2 + Módulo wsgi

- apache2 + gunicorn
- apache2 + uwsgi
- nginx + gunicorn
- nginx + uwsgi

Nota:

- **Tarea 1 (2 puntos)(Obligatorio)**: Documenta la instalación del módulo wsgi de apache2. Muestra los ficheros de configuración y muestra la ejecución del CMS.
- **Tarea 2 (2 puntos)(Obligatorio)**: Documenta la instalación y configuración de gunicorn y apache2. Muestra mezzanine funcionando y una comprobación de que, efectivamente, se está usando gunicorn.
- **Tarea 3 (2 puntos)(Obligatorio)**: Documenta la instalación y configuración de uwsgi y apache2. Muestra mezzanine funcionando y una comprobación de que, efectivamente, se está usando wusgi.
- **Tarea 4 (2 puntos)**: Documenta la instalación y configuración de gunicorn y nginx. Muestra mezzanine funcionando y una comprobación de que, efectivamente, se está usando gunicorn.
- **Tarea 5 (2 puntos)**: Documenta la instalación y configuración de uwsgi y nginx. Muestra mezzanine funcionando y una comprobación de que, efectivamente, se está usando wusgi.

10.10.1 Estudio de rendimiento

Ahora utilizando el script benchmark.py, realiza las pruebas de rendiemento para cada una de las configuraciones anteriores.

Nota:

- **Tarea 6 (5 puntos)**: Entrega la configuración del script de pruebas para cada una de las configuraciones. Entrega los datos obtenidos y la gráfica que has generado.

10.11 Práctica: Servidor de correos

Nota: (13 tareas - 35 puntos)(5 tareas obligatorias - 8 puntos)

Nota:

- Muestra al profesor: Tareas 3,4,5,6,7,8,9,10

Esta tarea consiste en instalar y configurar un servidor de correo similar al de cualquier organización, capaz de enviar y recibir directamente correo, almacenar los usuarios en LDAP, filtrar el correo en busca de virus o spam y servirlo a sus usuarios a través de los protocolos POP, IMAP y configurar un Webmail. Objetivos

1. Instalar y configurar un servidor postfix en un equipo con dirección IP pública dinámica
2. Aprender a configurar todos los componentes de un servidor de correos completo
3. Depurar el funcionamiento de un servicio

4. Documentar adecuadamente todo el proceso

10.11.1 Pasos a realizar en clase

Vamos a realizar un sistema de correo para el dominio tudominio.gonzalonazareno.org, cuyo servidor DNS lo administras en tu propio servidor DNS. Tienes que comunicar el nombre de dominio al profesor para configurar el servidor de correos del departamento. Instala postfix y comprueba que recibe correo directamente desde un equipo de Internet (hotmail, gmail, etc.). Configura tu servidor de correos para que use a babuino como relay y comprueba que puees enviar correos.

Nota:

- **Tarea 1 (1 puntos)(Obligatorio)**: Docuementa en redmine una prueba de funcionamiento, donde envíes desde tu servidor local al exterior. Muestra el log donde se vea el envío. Muestra el correo que has recibido.
- **Tarea 2 (1 puntos)(Obligatorio)**: Documenta en redmine una prueba de funcionamiento, donde envíes un correo desde el exterior(gmail, hotmail,...) a tu servidor local. Muestra el log donde se vea el envío. Muestra cómo has leido el correo.

Instala y configura un servidor dovecot POP e IMAP en tu equipo. Configura adecuadamente un cliente de correo (evolution, outlook, thunderbird, ...) para que reciba el correo a través de POP o IMAP. El cliente debe estar configurado en una máquina cliente. Nombra en tu servidor DNS al servidor smtp, pop e imap.

Nota:

- **Tarea 3 (2 puntos)(Obligatorio)**: Docuementa en redmine una prueba de funcionamiento, donde envíes desde tu cliente de correos al exterior. ¿Cómo se llama el servidor para enviar el correo? (Muestra la configuración).
- **Tarea 4 (2 puntos)(Obligatorio)**: Documenta en redmine una prueba de funcionamiento, donde recibas un correo desde el exterior(gmail, hotmail,...) y lo leas en tu cliente de correo. Utiliza el protocolo POP. ¿Cómo se llama el servidor para enviar el correo? (Muestra la configuración). Muestra una prueba de funcionamiento de cómo funciona el protocolo POP.
- **Tarea 5 (2 puntos)(Obligatorio)**: Documenta en redmine una prueba de funcionamiento, donde recibas un correo desde el exterior(gmail, hotmail,...) y lo leas en tu cliente de correo. Utiliza el protocolo IMAP. ¿Cómo se llama el servidor para enviar el correo? (Muestra la configuración). Muestra una prueba de funcionamiento de cómo funciona el protocolo IMAP.

Instala un webmail (roundcube) para gestionar el correo del equipo mediante una interfaz web. Instala y configura correctamente un sistema de filtrado de virus y spam utilizando amavis, clamav y spamassasin.

Nota:

- **Tarea 6 (3 puntos)**: Muestra al profesor el envío y recepción de correos utilizando el webmail.
- **Tarea 7 (5 puntos)**: Muestra al profesor el funcionamiento del sistema de filtrado de virus y spam.

10.11.2 Pasos a realizar en casa

- Configura adecuadamente el router de casa para que el puerto 25/tcp de tu equipo sea accesible desde Internet (eso se denomina DNAT o port forwarding)
- Date de alta en un servidor DNS dinámico como dyndns.org, no-ip.com, etc. o usa el nombre de dominio propio.

- Configura el DNS de tu proveedor para que la máquina a la que apunta el registro MX corresponda a tu IP pública. Si vas a utilizar un servicio gratuito como dyndns.org, no-ip.com, simplemente debes configurarlo para que apunte a tu ip.
- Instala postfix en tu máquina y comprueba que recibe correo directamente desde un equipo de Internet (hotmail, gmail, etc.)
- Prueba a enviar desde tu equipo un correo electrónico a jose@gonzalonazareno.org, que no lo rechazará aunque venga de una dirección IP dinámica.
- Prueba a enviar desde tu equipo un correo electrónico a hotmail/gmail. Comprueba si llega bien, si lo mete en SPAM o si rebota los mensajes (mira en /var/log/mail.log), ya que no acepta correos de direcciones IP dinámicas.
- Configura postfix para que envíe el correo electrónico a través de gmail como se indica en la documentación. Cuando funcione envía un correo a josedom24@gmail.com

---

Nota:

- **Tarea 8 (2 puntos)**: Envía el correo a jose@gonzalonazareno.org
- **Tarea 9 (3 puntos)**: Responde al correo que yo te voy a mandar desde esa dirección.
- **Tarea 10 (4 puntos)**: ¿Te rebota el correo enviado al exterior por qué estas usando ip dínamica? Independientemente de la respuesta, muestra el log donde se vea el envío de ese correo y documenta la configuración del relay con gmail. Finalmente envía un correo a josedom24@gmail.com.

10.11.3 Tarea adicional: Configuración de usuarios virtuales con LDAP

En el servidor de clase, configura postfix para que use usuarios virtuales guardados en un servidor ldap.

Instala un esquema adecuado para usuarios de postfix en LDAP y crea un script que reciba un nombre de usuario y añade un nuevo registro al LDAP:

1. El dn debes ajustarlo a la base a la de tu directorio
2. Cada entrada incluye un objectClass y atributos adecuados para postfix
3. El atributo mail es del tipo usuario@dominio
4. El buzón de cada usuario está en formato Maildir
5. El atributo userPassword es un hash SSHA del uid del usuario

---

Nota:

- **Tarea 11 (5 puntos)**: Documenta en redmine la configuración realizada. Y realiza una prueba de funcionamiento al profesor.

10.11.4 Tarea adicional: Configuración de seguridad para SMTP, POP e IMAP

En el servidor de clase, configura postfix para que las conecciones al servidor SMTP, POP e IMAP sean seguras (SSL).

Nota:

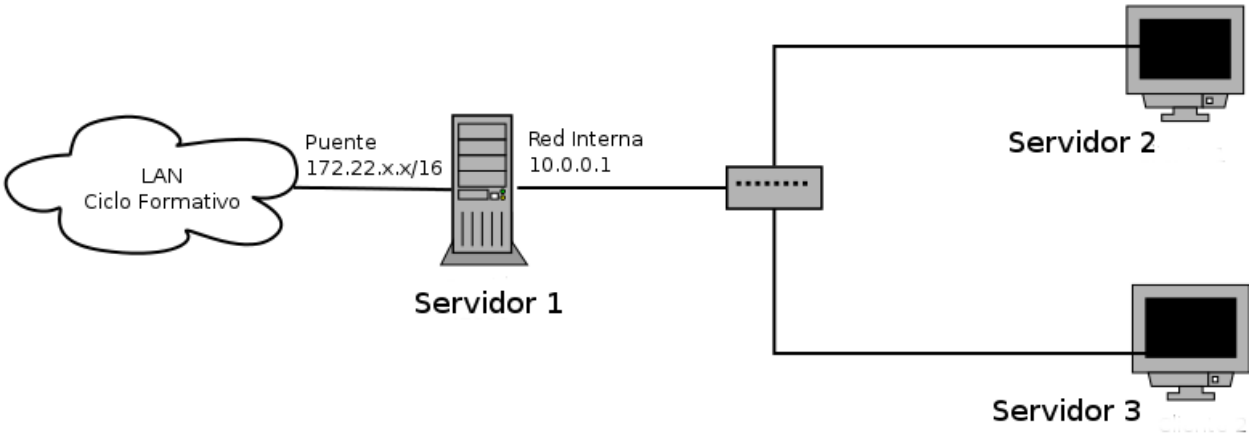
- **Tarea 12 (2 puntos)**: Documenta en redmine la configuración realizada para que nuestro servidor SMTP sea seguro. Indica alguna prueba de funcionamiento.

- **Tarea 13 (3 puntos)**: Documenta en redmine la configuración realizada para que nuestro servidor POP o IMAP sea seguro. Indica alguna prueba de funcionamiento.

10.12 Práctica: Servidor proxy-cache, proxy inverso y balanceador de carga

Nota: (16 tareas - 30 puntos)(8 tareas obligatorias - 15 puntos)

En primer lugar, construye con KVM o con vagrant la siguiente infraestructura:



10.12.1 Proxy squid

Queremos instalar un servidor proxy/cache en nuestro **servidor 1**. Con ello vamos a poder controlar las páginas web a las que accedamos (desde el **servidor 2** y **servidor 3**), además de acelerar nuestra navegación.

Nos piden la configuración de un proxy/cache/filtro en nuestra infraestructura. Hemos elegido como proxy/cache squid3, y como filtro de contenido dansguardian. Tenemos que tener en cuenta las siguientes consideraciones:

1. El proxy/cache sólo admite conexiones de la red local.
2. Se quieren limitar las siguientes conexiones:
   - No se pueden bajar ficheros que se puedan instalar (`exe,msi,rar,zip,bin,iso`).
   - No tienen acceso a internet los fines de semana.
3. El control de las páginas permitidas se hará mediante listas negras usando dansguardians.
4. Por último tendremos instalado un programa para monitorizar el uso del proxy: sarg. Para visualizar la información generada por dicho programa accederemos a una página web llamada `proxy.josedomingo.gonzalonazareno.org` que sólo será accesible si ponemos el nombre de usuario y contraseña.
5. Finalmente queremos configurar la infraestrucutra para tener un proxy transparente.

Nota:

- **Tarea 1 (1 punto)(Obligatorio)**: Configura de forma manual el proxy. Muestra las capturas de pantalla.

- **Tarea 2 (2 puntos)(Obligatorio)**: Muestra la configuración de squid para no permitir descargar ficheros ejecutables. Prueba de funcionamiento.
- **Tarea 3 (2 puntos)(Obligatorio)**: Muestra la configuración de squid para no permitir el acceso los fines de semana. Prueba de funcionamiento.
- **Tarea 4 (2 puntos)**: Filtra el dominio youtube.com en la lista negra y prueba que realmente no se puede acceder.
- **Tarea 5 (2 puntos)**: Documenta la instalación de sarg, y muestra las estadísticas de acceso al proxy con sarg.
- **Tarea 6 (3 puntos)**: Documenta la configuración del proxy transparente y haz una prueba de funcionamiento.

10.12.2 Proxy inverso

Nota: Seguimos trabajando con las mismas máquinas, pero en un ejercicio nuevo, por lo que si necesitas detener los servicios del ejercicio anterior lo puedes hacer.

En este caso queremos instalar dos servidor web en el **Servidor 2** y en **Servidor 3**, estos servidores deben servir una web completa (con hoja de estilo, imágenes,...) busca alguna plantilla.

En el **Servidor 1** vamos a instalar diferentes configuraciones de proxy inverso para que desde el exterior se puedan acceder a las páginas de los servidores conectados a la red interna. Los proxy inversos los vamos a configurar de dos maneras distintas:

- Para que se acceda al servidor de **Servidor 2** con la URL `www.servidor.org\pagina1` y al servidor **Servidor 3** con la URL `www.servidor.org\pagina2`.
- Para que se acceda al servidor de **Servidor 2** con la URL `www.pagina1.org` y al servidor **Servidor 3** con la URL `www.pagina2.org`.

Nota:

- **Tarea 7 (2 puntos)(Obligatorio)**: Configura apache2 como proxy inverso para acceder a los servidores internos de la primera forma.
- **Tarea 8 (2 puntos)**: Configura apache2 como proxy inverso para acceder a los servidores internos de la segunda forma.
- **Tarea 9 (2 puntos)(Obligatorio)**: Configura nginx como proxy inverso para acceder a los servidores internos de la primera forma.
- **Tarea 10 (2 puntos)**: Configura nginx como proxy inverso para acceder a los servidores internos de la segunda forma.

10.12.3 Balanceador de carga

Nota: Seguimos trabajando con las mismas máquinas, pero en un ejercicio nuevo, por lo que si necesitas detener los servicios del ejercicio anterior lo puedes hacer.

Ajustar la configuración de las dos máquinas del cluster de balanceo (**Servidor 2** y **Servidor3**):

Deshabilitar la opción `KeepAlive` en el fichero de configuración `/etc/apache2/apache2.conf` para realizar la evaluación del rendimiento sin la opción de reutilización de conexiones, para ello en `/etc/apache2/apache2.conf`:

```
...
KeepAlive Off
...
```

Advertencia: Nota: este ajuste no es estrictamente necesario (y sería desaconsejable en un entorno de producción real), pero facilita las pruebas manuales dado que permite detectar inmediatamente el "cambio" de destino resultado del balanceo de carga manteniendo la opción por defecto, en las pruebas manuales desde el navegador sería necesario esperar 5 segundos (el time out de keep alive) antes de recargar la página y ver el efecto del reparto de carga

En el **Servidor 1** vamos a realizar diferentes configuraciones de servicios para que realicen el balanceo de carga entre los servidores web internos, por lo tanto al acceder desde el exterior a la ip del **Servidor 1** se irá mostrando alternativamente las páginas de **Servidor 2** y **Servidor 3**.

Nota:

- **Tarea 11 (2 puntos)(Obligatorio)**: Configura apache2 como balanceador de carga.
- **Tarea 12 (2 puntos)(Obligatorio)**: Configura ngninx como balanceador de carga.

Balanceo de carga con haproxy

Instala haproxy en **Servidor 1**. Lo primero es configurar HAproxy en balanceador (de momento sin soporte de sesiones persistentes)::

```
servidor1:~# cd /etc/haproxy
servidor1:/etc/haproxy/# mv haproxy.cfg haproxy.cfg.original
servidor1:/etc/haproxy/# nano haproxy.cfg

global
    daemon
    maxconn 256
    user    haproxy
    group   haproxy
    log     127.0.0.1       local0
    log     127.0.0.1       local1  notice

defaults
    mode    http
    log     global
    timeout connect 5000ms
    timeout client  50000ms
    timeout server  50000ms

listen granja_cda
    bind 193.147.87.47:80
    mode http
    stats enable
    stats auth  cda:cda
```

(continues on next page)

(proviene de la página anterior)

```
    balance roundrobin
    server uno 10.0.0.X:80 maxconn 128
    server dos 10.0.0.Y:80 maxconn 128
```

Define (en la sección listen) un «proxy inverso» de nombre `granja_cda` que:

- trabajará en modo http (la otra alternativa es el modo tcp, pero no analiza las peticiones/respuestas HTTP, sólo retransmite paquetes TCP)
- atendiendo peticiones en el puerto 80 del balanceador
- con balanceo round-robin
- que repartirá las peticiones entre dos servidores reales (de nombres uno y dos) en el puerto 80 de las direcciones 10.0.0.X y 10.0.0.Y
- adicionalmente, habilita la consola Web de estadísticas, accesible con las credenciales `cda:cda`

Más detalles en Opciones de configuración HAPproxy 1.5

Para iniciar HAproxy es necesario habilitar en `/etc/default/haproxy` el arranque de HAproxy desde los scripts de inicio, estableciendo la variable `ENABLED=1`

Desde la máquina cliente abrir en un navegador web la URL `http://172.22.x.x` y recargar varias veces para comprobar como cambia el servidor real que responde las peticiones.

Advertencia: Nota: Si no se ha deshabilitado la opción KeepAlive de Apache, es necesario esperar 5 segundos entre las recargas para que se agote el tiempo de espera para cerrar completamente la conexión HTTP y que pase a ser atendida por otro servidor.

Desde la máquina cliente podemos abrir en un navegador web la URL `http://172.22.x.x/haproxy?stats` para inspeccionar las estadísticas del balanceador HAProxy (pedirá un usuario y un password, ambos `cda`).

Nota:

- **Tarea 13 (2 puntos)(Obligatorio)**: Muestra al profesor y entrega capturas de pantalla que el balanceador está funcionando.
- **Tarea 14 (1 punto)**: Entrega una captura de pantalla donde se vea la página web de estadísticas de haproxy.
- **Tarea 15 (1 punto)**:Desde uno de los servidores (**Servidor 2** ó **Servidor 3**), verificar los logs del servidor Apache. En todos los casos debería figurar como única dirección IP cliente la IP interna de la máquina balanceador [10.0.0.1]. ¿Por qué?

Configurar la persistencia de conexiones Web (sticky sessions)

Vamos a añadir las opciones de persistencia de conexiones HTTP (sticky cookies) al fichero de configuración. Para ello vamos a modificar las tres últimas líneas del fichero de configuración:

```
    ...
    cookie PHPSESSID prefix
    server uno 10.0.0.X:80 cookie EL_UNO maxconn 128
    server dos 10.0.0.Y:80 cookie EL_DOS maxconn 128
```

El parámetro `cookie` especifica el nombre de la cookie que se usa como identificador único de la sesión del cliente (en el caso de aplicaciones web PHP se suele utilizar por defecto el nombre `PHPSESSID`). Para cada «servidor real» se especifica una etiqueta identificativa exclusiva mediante el parámetro `cookie`. Con esa información HAproxy reescribirá las cabeceras HTTP de peticiones y respuestas para seguir la pista de las sesiones establecidas en cada «servidor real» usando el nombre de cookie especificado (PHPSESSID):

- conexión cliente -> balanceador HAproxy : cookie original + etiqueta de servidor
- conexión balanceador HAproxy -> servidor : cookie original

En los servidores web **Servidor 2** y **Servidor 3** vamos a configurar apache2 para que puedan ejecutar php y vamos a crear el fichero `sesion.php` con el siguiente contenido:

```
<?php
     header('Content-Type: text/plain');
     session_start();
     if(!isset($_SESSION['visit']))
     {
             echo "This is the first time you're visiting this server";
             $_SESSION['visit'] = 0;
     }
     else
             echo "Your number of visits: ".$_SESSION['visit'];

     $_SESSION['visit']++;

     echo "\nServer IP: ".$_SERVER['SERVER_ADDR'];
     echo "\nClient IP: ".$_SERVER['REMOTE_ADDR'];
     echo "\nX-Forwarded-for: ".$_SERVER['HTTP_X_FORWARDED_FOR']."\n";
     print_r($_COOKIE);
?>
```

Para realizar la comprobación de que la sesión se mantiene aunque estemos balanceando, en la máquina cliente, arrancar el sniffer de red whireshark y ponerlo en escucha sobre el interfaz eth0 (fijar como filtro la cadena http para que solo muestre las peticiones y respuestas HTTP).

- desde el navegador web del cliente acceder varias veces a la URL `http://172.22.x.x/sesion.php` (comprobar el incremento del contador [variable de sesión])
- acceder la misma URL desde otro navegador (o desde una pestaña de «incógnito») para forzar la creación de una nueva sesión:

Detener la captura de tráfico en wireshark y comprobar las peticiones/respuestas HTTP capturadas.

Nota:

- **Tarea 16 (2 puntos)**:Verificar la estructura y valores de las cookies PHPSESSID intercambiadas. En la primera respuesta HTTP (inicio de sesión), se establece su valor con un parámetro HTTP SetCookie en la cabecera de la respuesta. Las sucesivas peticiones del cliente incluyen el valor de esa cookie (parámetro HTTP Cookie en la cabecera de las peticiones)
